Universiteit Utrecht

UTRECHT UNIVERSITY

MSc THESIS PROJECT
IN GAME AND MEDIA TECHNOLOGY PROGRAM

# An Entropy Model for Identifying Loitering Behavior in Videos

*Maguell L.T.L. Sandifort*
*ICA-3898903*

supervised by
Dr. Wolfgang Hürst
Dr. Jianquan Liu

second examiner:
Professor Dr. Remco Veltkamp

This thesis has been done as part of an internship at NEC Japan research labs.

November 29, 2018

## SUMMARY

This thesis addresses the problem of how loitering behavior in public spaces can be detected automatically from the footage of surveillance cameras. This is an important problem, because loitering often leads to criminal behavior. Automatic detection is necessary, because the large amount of data requires tools that pre-filter potential loitering cases and support humans in deciding if necessary actions have to be taken.

The research of this thesis has been done as part of an internship at NEC System Platform Research Laboratories[1] in Kawasaki, Japan. It was supervised on location by Dr. Jianquan Liu, from NEC labs, and remotely by Dr. Wolfgang Hürst from Utrecht University. The research is a continuation of NEC's existing work in the area of loitering behavior detection. The demo system that has been built as part of this thesis was implemented on top of the existing tools from NEC. All other implementations and research has been done from scratch with only minor inclusions of existing parts or knowledge. The scientific results have already been published at different scientific events, in particular:

- A full paper at ICMR 2018: Maguell L. T. L. Sandifort, Jianquan Liu, Shoji Nishimura, and Wolfgang Hürst. 2018. An Entropy Model for Loiterer Retrieval across Multiple Surveillance Cameras. In Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval (ICMR '18). ACM, New York, NY, USA, 309-317. DOI: https://doi.org/10.1145/3206025.3206049
- A reviewed demonstration paper at ICMR 2018: Maguell L.T.L. Sandifort, Jianquan Liu, Shoji Nishimura, and Wolfgang Hürst. 2018. VisLoiter+: An Entropy Model-Based Loiterer Retrieval System with User-Friendly Interfaces. In Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval (ICMR '18). ACM, New York, NY, USA, 505-508. DOI: https://doi.org/10.1145/3206025.3206091
- A poster presented at the Japanese domestic conference VC 2018: Maguell L.T.L. Sandifort, Jianquan Liu, Shoji Nishimura, Wolfgang Hürst. 2018. Loiterer Retrieval and Visualization using Entropy Model. VC 2018, Yamagata, Japan, P14. http://cgvi.jp/vc2018/program/poster/

The full paper published at ICMR 2018 is the main outcome of this thesis. ICMR is "a premier conference to display scientific achievements and innovative industrial products in the field of multimedia retrieval"[2] and a major event in the international multimedia retrieval community. The research presented in the paper has also been demonstrated at the event via the accompanying demo paper and as a poster at the Japanese domestic conference Visual Computing 2018. This report summarizes the scientific results of the thesis and internship and contains the following items:

- The full paper published at ICMR 2018
- The ICMR 2018 demo paper
- The extended abstract of the VC 2018 poster

All necessary information about the research is contained in the ICMR papers. Some details about the data could not have been published here due to confidentiality issues and non-disclosure agreements with NEC.

[1]NEC Biometrics Research Laboratories as of April 2018

[2]Quote from the ICMR 2018 website, http://www.icmr2018.org/index.html

# Acknowledgements

I would like to thank Dr. Wolfgang Hürst for his guidance, support and patience as supervisor at UU. Next I want to thank Dr. Jianquan Liu for his advice and guidance and without whom, this project would not have been possible. My thanks also go to the NEC System Platform team for helping in everyday life and making me feel at home. Finally, my thanks to NEC, for giving me the opportunity to do my thesis at their research labs and providing the funds for my stay in Japan.

# Contents

# Chapter 1

# Scientific Paper (published at ICMR 2018)

# An Entropy Model for Loiterer Retrieval across Multiple Surveillance Cameras

## ABSTRACT

Loitering is a suspicious behavior that often leads to criminal actions, such as pickpocketing and terrorist attacks. 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 a substitution for 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.

## 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 [16]. It is seen as a precursor to illegal behavior such as pickpocketing or illegal entry.

A number of approaches exist for detecting loitering behavior [7]. 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 [6, 13].

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. While we could apply single-camera methods to all these cameras in parallel this would treat each camera as their own confined space and we will not get any insight into the big picture of the entire surveilled scene.

[10] addresses 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 and provides a list of loitering candidates based on the amounts of facial features captured of one individual, which signifies that they have 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's 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 re-appearance to provide a loitering candidate list.

An overview of our entropy model can be seen in Figure 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 in side 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:

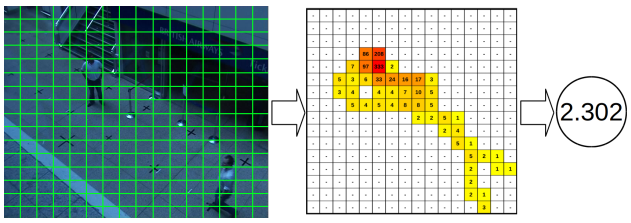


**Figure 1: Overview of the entropy model. Left: Split the camera view up into bins. Middle: Add each feature to the correct bin based on location in camera view. Right: Calculate the entropy score**

*ICMR'18, June 11–14, 2018, Yokohama, Japan*
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM. . . $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

- 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 pontential loitering candidates.
- We conduct experiments over several datasets to evaluate our method and compare it to other loitering 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 can provide additional interesting candidates.

## 2 RELATED WORK

One way to detect loitering behavior is by looking at the trajectory of a pedestrian.

Ko et al. [8] 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. [9] 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. [14] 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 micro-approaches require consistent 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.

Some methods utilize entropy theory to detect abnormalities.

Ren et al. [15] propose a behavior entropy model to detect abnormal behaviors in a crowd.

Xiong et al. [18] 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. This 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. We will apply entropy theory to individuals as a measure to determine how much they are moving.

In Bird et al. [3] a single camera method is used to detect loiterers around a bus stop. Their method takes snapshots of people that are clearly visible and then use the clothing color of people for similarity matching. Loitering is based on the duration of appearance.

Tomas et al. [17] identifies sequential micro-patterns (such as walk, stop, walk, stop) to determine loitering behavior of the elderly.

Elhamod et al. [4] provides 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 [13] 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.

Huang et al. [6] 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. [2] uses multiple cameras surveilling the same scene at different angles to track and detect loiterers and abandoned luggage.

Lu et al. [12] 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 previous 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 that move and people that are waiting. When analyzing a large amount of footages, 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 way the operator can make the final judgement. Our method can retrieve loiterers and provide other suspicious candidates in an ordered list sorted by suspicion level.

**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 |

## 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 loirtering 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. 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.

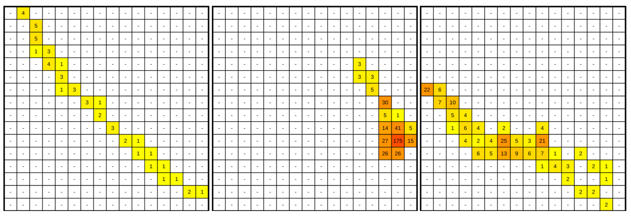


**Figure 2: Heatmap examples showing the abstractions of threee different movements of a person: left) walk through the area; middle) keep waiting in the area; right) walk around many times, likely loitering in the area.**

Some heatmap examples can be seen in Figure 2. To the left, it 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. In the middle, it is an example of someone who is waiting. The moving has a very skewed distribution: in the location where the waiting is shown a high peak. On the right, it is an example of someone who is loitering by walking, stopping, changing direction, and repeat. The person does not move in a straight line and therefore is more spread out over the heatmap.

To obtain the movement information, we are inspired by the entropy theory [5] to model the latent information in the heatmap. To do this, 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.

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

Where, the empty bins do not contribute to the entropy, thus they are excluded from the computation.

If someone is moving around on the camera view, more bins in the heatmap will be filled causing the entropy to rise. In addition, if someone is moving, the distribution is equalizing over the bins, causing the entropy to rise. Finally, if someone is standing still, the distribution of the heatmap will be more skewed, causing the entropy to drop.

Figure 3 shows the entropy $WES$ 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,

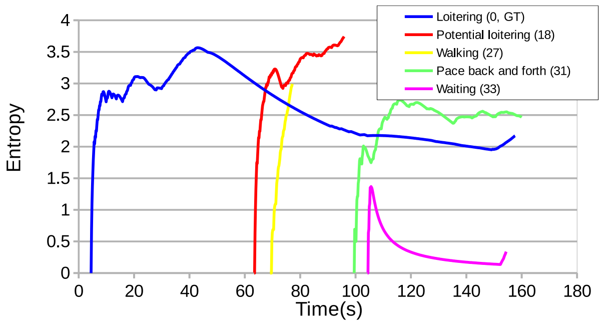


**Figure 3: WES of five persons over time. If someone is moving, the WES will increase, while if standing still the WES will decrease. WES of going up and down shows the move, stop, move, stop behavior. Here, GT means the groundtruth of loiter in the dataset. The legend is associated with an id of a person extracted from the dataset.**

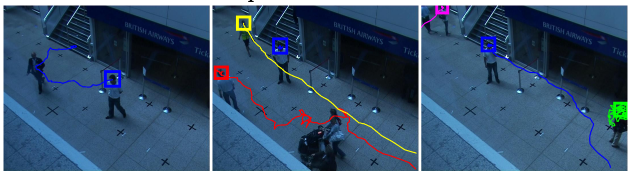


**Figure 4: An example selected from the dataset (PETS S1) corresponding to Fig. 3. It shows frames at the times: 40s, 70s and 110s. In addition, the path is drawn that they will make until the next image. For the last image, the path is drawn until they leave the scene.**

0) starts by pacing through the area causing 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 loitering 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 candidate together into the weighted entropy score $WES$.

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

**Definition** 2 (**WES**: Weighted Entropy Score). *Given a sequence of $CA$s, 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.*

This way, longer $CA$s are weighted more substantially. Short $CA$s with high entropy, which will be someone walking through a camera's view, will not contribute as much as longer longer $CA$s.

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 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 $CA$s 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. We want to distinguish between this as well, therefore we count each reappearance on a camera as more substantial than the previous one.

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

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

where $a$ is the amount of $CA$s for that person on camera $i$. This way, each reappearance on the same camera will be counted more heavily than the last.

Finally, we want to normalize 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 move repeatedly leave and enter the camera's view. With this behavior the pickpockets's $RAS$ will be high.

**(3) Duration.** A person who is in an area for a longer period of time is more likely to be loitering. Therefore, we want 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 $CA$s of a candidate together. In the case that $CA$ duration 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.

*Loitering Score.* Each of the measures above account for an aspect of loitering behavior. To provide a list of candidates, we need a way to determine how likely someone is loitering. For this 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.

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

Each factor can be scaled by its associating weight $\alpha$, $\beta$ and $\gamma$. As the score is a comparison value, these do not have to add up to 1, only the ratio between the three should be considered.

The Loitering Score 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 1. 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:** BuildCandidateList()

```
Input: Data set $ds$ with features $f_i^j$ grouped by person
       $j$, RAS scalar $\alpha$, WES scalar $\beta$, Duration scalar
       $\gamma$, heatmap Dimension $hD$
Output: List of candidates $CL$ ordered by score.
1  begin
2      foreach Candidate $j \in db$ do
3          $CAL^j \leftarrow$ GroupAppearances() ;
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         $LS \leftarrow j$ ;
12     Sort candidate list by $LS$ ;
13     return Ordered candidate list ;
```

*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.

As we calculate our measure per person, we first group the features by person identifier. For each person, Algorithm 2 then groups the features into $CA$s (line 1.3).

First the features are sorted by timestamp so they are in chronological order (line 2.2). As people can appear simultaneously on multiple cameras, we keep track of open cameras. If we add metadata to an open camera, we compare the timestamp of the last metadata added to that $CA$ Each time we add a feature to a $CA$, we compare the timestamp of the last feature added to that $CA$ (line 2.5). If the time difference exceeds the $T_{ca}$, we push the old $CA$ to the list and add the feature to a new $CA$ for the same camera. (line 2.6 and 2.7). Otherwise, we just add the feature to the existing $CA$ (line 2.8).

Finally, after we have added all features to a $CA$ we push the remaining $CA$s to the list (line 2.9 and 2.10) and then return the list of $CA$s (line 2.11). We then use this list of $CA$s for that person to calculate the measures.

**Algorithm 2:** 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 $CAL^j$ ;
```

*Measures.* For each $CA$ we create a heatmap. We split the heatmap into bins according to the $hD$. (line 1.5) For a $hD$ of 7, we split the heatmap into 7x7 bins, for a total of 49 bins. Then we iterate over the features in the $CA$. By using the position metadata of the feature, we determine its center coordinates and map that to the appropriate bin in the heatmap. Next we calculate the entropy of each heatmap (line 1.6).

After that, we have the information we need to calculate the measures $WES$, $RAS$ and $DS$(line 1.7, 1.8 and 1.9). Then we calculate the $LS$ by scaling each measure with their appropriate scalar (line 1.10), after which the processing for that candidate is done. We then push the candidate $j$ to the Candidate List $CL$ (line 1.11) and continue to next person.

After all people have a score assigned to them, we order the candidate list on the $LS$ (line 1.12).

## 4 EXPERIMENTS

We evaluate our approach by using two open reald datasets.

- AntiLoiter Dataset[1] [1], and
- PETS 2007[2].

*AntiLoiter Dataset.* A multi-scene multi-camera dataset contains 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.

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

*PETS2007 Dataset.* Single-scene multi-camera footage of an airport. The cameras have a resolution of 768x576. Consists of multiple scenes for the detection of loitering and abandoned luggage. To compare our approach to one of the state of the art [13] and to show that we can use our approach on single camera scenes. We only use the third 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. This way we receive the features in a similar way as an accurate tracking method that can deal with occlusions. We use the datasets scenario 1 and 3 for our comparison. Scenario 1 is a scene that shows one person loitering in the scene. Scenario 3 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 this 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 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 as even the smallest of movements will cause a person that is waiting 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 the PETS2007 dataset by varying the bin dimensions in different sizes.

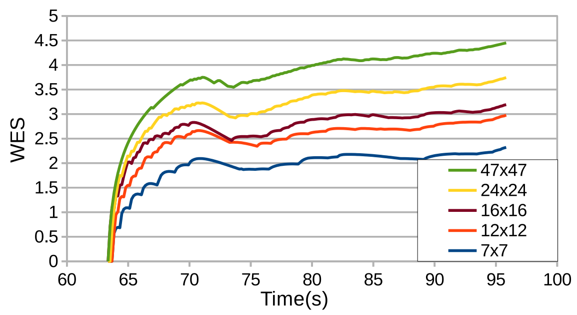


**Figure 5: Effect of the Heatmap Dimensions on the WES of a loitering candidate over time. The $WES$ increases with heatmap dimension size from 7x7 to 47x47.**

Figure 5 shows the WES of a loiterer over time for different Heatmap Dimensions. As shown in the Figure, the WES increases steadily with the dimension size, such as from 7x7 to 47x47.

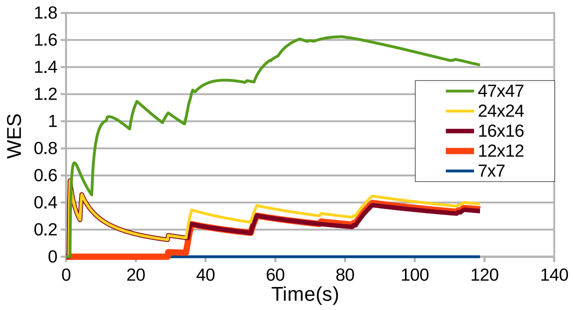


**Figure 6: Effect of the Heatmap Dimensions on the WES on a waiting person. Low heatmap dimensions will cause the $WES$ to be 0.**

Figure 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 cell to be filled with features, which results in an entropy of 0. Therefore we want to take heatmap 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: $\alpha$ for the $RAS$, $\beta$ for the $WES$ and $\gamma$ 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 Figures 7, 8 and 9.

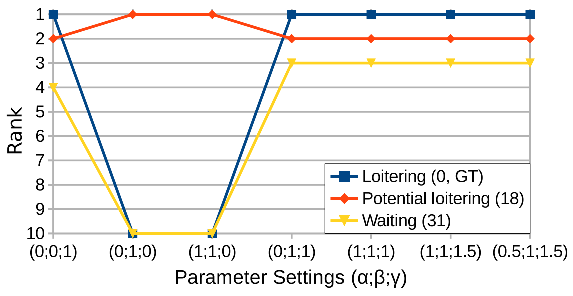


**Figure 7: Effect of the scaling parameters on the ranking of several candidates from the PETS S1 dataset.**

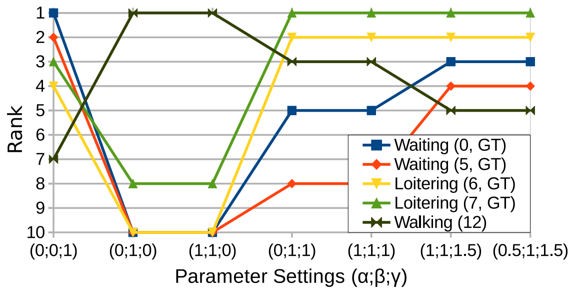


**Figure 8: Effect of the scaling parameters on the ranking of several candidates from the PETS S3 dataset**

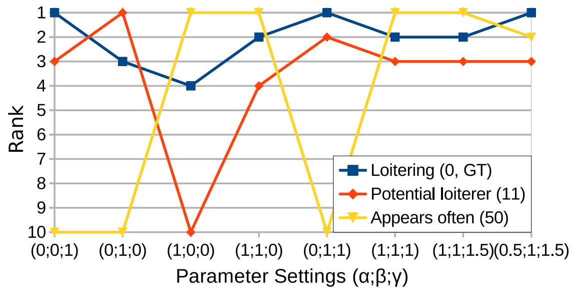


**Figure 9: Effect of the scaling parameters on the ranking of several candidates from the AntiLoiter dataset**

As can be seen in the above figures, only using the duration (0;0;1) presents cand

Using only the duration like other methods do, we already get reasonable results for PETS2007 and AntiLoiter.

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 Figures 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 Figure 7, people that are just briefly walking through the area are now scoring higher than those waiting for a very long time. In the PETS2007 dataset the ground truth contains two people that are waiting as well, thus we increase the weight of the duration. With this, the results can be retrieved pretty well.

## 4.2 Comparison with Related Work

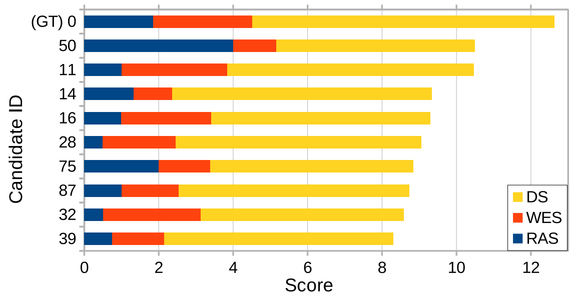


**Figure 10: Top 10 candidates on the AntiLoiter Dataset**

We compared our method to two states of the art [11, 13]. The following parameters were used: $\alpha = 0.5$, $\beta = 1.0$, $\gamma = 1.5$, Heatmap dimensions: 16x16.

Figures 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 requires a loiter time threshold to provide 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 Figure 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 $\beta$ 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.

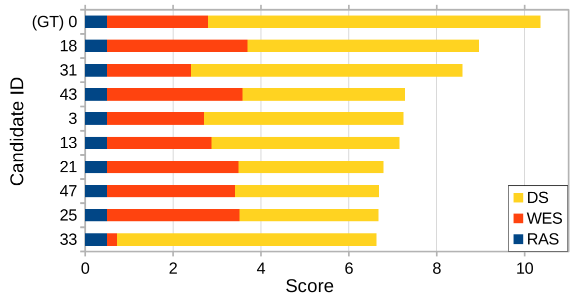


**Figure 11: Top 10 candidates on the PETS2007 Dataset, Scenario 1**

In the comparison with PETS2007 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 Figure 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.

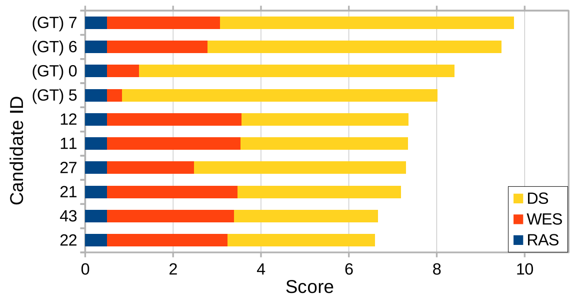


**Figure 12: Top 10 candidates on the PETS2007 Dataset, Scenario 3**

In the comparison with PETS2007 S3 Scenario 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 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 figure 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.

**Table 2: Comparison of the results**

| | *Datasets:* | | AntiLoiter | PETS | |
|---|---|---|---|---|---|
| *Method* | | | | S1 | S3 |
| MM16 | | P | 100% | - | - |
| | | R | 100% | - | - |
| MTA15 | | P | - | 99.8% | 100% |
| | | R | - | 96.22% | 58.85% |
| Proposed | Top 1 | P | **100%** | **100%** | 100% |
| | | R | **100%** | **100%** | 25% |
| | Top 4 | P | 25% | 25% | **100%** |
| | | R | 100% | 100% | **100%** |
| | Top 10 | P | 10% | 10% | 40% |
| | | R | 100% | 100% | 100% |

## 5 DISCUSSION AND FUTURE WORK

We did not have an optimal dataset to test our system 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 ir 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] [n. d.]. AntiLoiter Dataset. http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/. ([n. d.]). Last Accessed: 2018-01-19.

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

[3] 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.

[4] 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.

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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

[11] 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

[12] 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.

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

[14] 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.

[15] 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.

[16] 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

[17] 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.

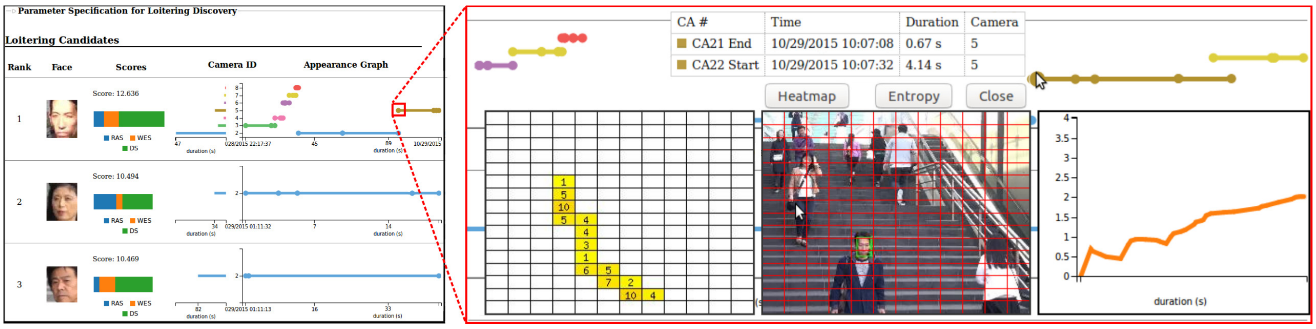
[18] Guogang Xiong, Xinyu Wu, Yen-lun Chen, and Yongsheng Ou. 2011. Abnormal crowd behavior detection based on the energy model. In *Information and Automation (ICIA), 2011 IEEE International Conference on*. IEEE, 495–500.

# Chapter 2

# Demo Paper (published at ICMR 2018)

# VisLoiter$^+$: An Entropy Model-Based Loiterer Retrieval System with User-friendly Interfaces

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



| CA # | Time | Duration | Camera |
|---|---|---|---|
| CA21 End | 10/29/2015 10:07:08 | 0.67 s | 5 |
| CA22 Start | 10/29/2015 10:07:32 | 4.14 s | 5 |

**Figure 1: System overview and interfaces of VisLoiter$^+$. (left) A ranked list of loiterer candidates retrieved by VisLoiter$^+$, where each row presents the abstracted information of a person including a face thumbnail, integrated scores for ranking, a histogram of appearing duration w.r.t different cameras, and an appearance graph showing how the transition of a person was appearing in different cameras. (right) When clicking on the point line of a camera, the details of a person will be shown in a popup window. There, at the top, the appearance information is summarized such as the start timestamp and end timestamp, the duration, and the camera id; in the middle, a lightweight playback of the video clip of the appearing person is displayed; by clicking on the "Heatmap" button, a grid will be overlaid on the video clip and a heatmap will be attached to show the distribution of features of the candidate; by clicking on the "Entropy" button, an entropy graph will be attached to show the change of entropy w.r.t the heatmap over time.**

## ABSTRACT

It is very difficult to fully automate the detection of loitering behavior in video surveillance, therefore humans are often required for monitoring. Alternatively, we could provide a list of potential loiterer candidates for a final yes/no judgment of a human operator. Our system, VisLoiter$^+$, realizes this idea with a unique, user-friendly interface and by employing an entropy model for improved loitering analysis. Rather than using just frequency of appearance, we expand the loiter analysis with new methods measuring the amount of person movements across multiple camera views. The interface gives an overview of loiterer candidates to show their behavior at a glance, complemented by a lightweight video playback for further details about why a candidate was selected. We demonstrate that our system outperforms state-of-the-art solutions using real-life data sets.

## 1 INTRODUCTION

Nowadays, most public areas are equipped with a large amount of surveillance cameras to enhance security. One security risk is people that are loitering, i.e., staying in an area for an extended period of time without a clear goal [10]. Loitering is often followed up by illegal actions such as illegal entry or pickpocketing and is therefore an important behavior to detect. However, to do this, the camera footage needs to be viewed by operators. Manually searching through footages of a multi-camera system is time consuming and requires a lot of man power. While there are many automatic loitering detection approaches [1, 2, 4, 9, 12] that do not require an operator, they require a threshold to recognize loitering, leading to potential misclassification above or below this threshold line. Research suggests that operators will always be necessary in some form [3]. Therefore a suitable alternative to automated recognition is reducing the workload of operators instead of replacing them by providing an overview list of loiterer candidates. This eases the job of the operators, but still leaves the final yes/no judgment to them.

Liu et al. introduced AntiLoiter [5], which provides such a list of candidates based on the frequency of appearance. This state-of-the-art commercial product extracts identifiable features of people from video frames. After that the features are grouped by similarity to get clusters of features that are of the same person. However, frequency or duration is not the only indicator of loitering behavior. For example, the trajectory information of people is a considerable indicator, but not used in AntiLoiter. They opt to forgo tracking methods due to tracking requiring a continuous view, which is difficult to achieve in crowded scenes or across multiple cameras.

Liu et al. also provided a visualization system for their method named VisLoiter [6] that visualizes the discovery results of loiterer candidates. For each candidate they then show a chart of appearances. The higher the amount of features, the greater the segment will be. Clicking on one of the appearances plays back the video of those appearance segments. However, as the amount of features scales with the duration, which is already on the x-axis, it does not provide much additional information. Also, the overview does not display which camera view the features were captured from, so operators will have to open the video playback to find this out.

In this paper we propose an improved loiterer retrieval system that builds on the previous versions mentioned above (AntiLoiter [5], VisLoiter [6]). To achieve this goal, our new approach needs to focus on three aspects of loitering behavior by answering three questions. (1) How irregular is the abstracted movement of a person in a camera or across multi-cameras? (2) How frequent is the same person re-entering the same camera? (3) How abnormal is the duration of a person showing up on the same camera?

*ICMR'18, June 11–14, 2018, Yokohama, Japan*
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Differences between VisLoiter$^+$ and previous versions.

| **Functionality** | **VisLoiter$^+$** | **AntiLoiter** [5] **VisLoiter** [6] |
|---|---|---|
| List of candidates | ✓ | ✓ |
| Uses duration of appearance | ✓ | ✓ |
| Consider reappearance on cameras | ✓ | |
| Consider movement of candidates | ✓ | |
| Overview of duration per camera | ✓ | |
| Overview of appearances on cameras | ✓ | |
| Allow tuning for retrieval ranking | ✓ | |
| Allow video playback | ✓ | ✓ |

Therefore, responding to the three questions, we propose (1) employing an entropy model, which utilizes the more in-depth information that tracking approaches can provide, but without relying on detailed tracking. We map the positions of a person's features to a heatmap and calculate the entropy over it to determine how the person has moved across the camera view. This model serves as an abstraction for tracking methods. In addition, we consider (2) the transition of a person between cameras, as well as (3) the duration of appearance. These three aspects are then combined into a final score, which is used to determine the rank of loitering candidates.

Regarding the functionality and the interface, our new VisLoiter$^+$ provides an overview of the top loitering candidates, which an operator can use to make the final call. In addition, it gives an overview of which cameras candidates have appeared on and what has caused them to have their high loitering score. Despite these overviews, it is often still necessary to manually observe the video footage. Therefore, VisLoiter$^+$ allows operators to further investigate candidates of choice by showing the appearances of the candidate over time and playing video segments with the candidate in it. The operator can use this information to make the final call on whether or not someone is a loiterer.

Finally, for the purpose of demonstration, we will present a live demo where we highlight our improvements and show the usability of VisLoiter$^+$ with user-friendly interfaces. We also show that VisLoiter$^+$ can provide additional potential loitering candidates based on their amount of movement and reappearances on the same camera. We summarize our key contributions and differences to previous work in Table 1.

## 2 NEW APPROACH: VISLOITER$^+$

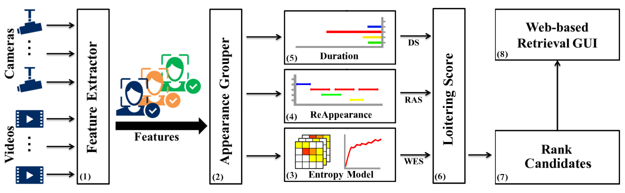


Figure 2: Architecture of VisLoiter$^+$.

Fig. 2 shows the architecture of VisLoiter$^+$. In the *Feature Extractor* (1), any feature extraction method can be used, as long as features from the same person can be grouped together, such as by means of a similarity function. The metadata of a feature includes: a person identifier, a camera identifier, a timestamp and the position in the camera view. Next the *Appearance Grouper* (2) groups features into Camera Appearances (*CA*s). Each *CA* contains features from one person on one camera where they were extracted. Then we calculate three loitering measures: the Weighted Entropy Score *WES* using our *Entropy Model* (3), the *ReAppearance Score RAS* (4) using the *CA*s, and a *Duration Score DS* (5). These measures are then weighted and combined into one (6) *Loitering Score LS* (6). Finally, the candidates are *Ranked* by their *LS* (7) and sent, along with their representation data, to the *Web-Based GUI* (8), where the operator can check the results.

### 2.1 Loitering Measures

To determine the likeliness of loitering behavior, we incorporate three different measures: entropy, reappearance and duration. Each of these measures contribute to the final score for retrieval ranking. Why we need to propose these measures is also corresponding to the questions we mentioned in Section 1.

First the metadata of the features are grouped into Camera Appearances (*CA*s). A *CA* starts when a person enters the camera view and ends when they leave. We provide a *CA* timeframe parameter $T_{CA}$, to determine how much time has to pass for 2 features to be considered in a different camera appearance. The *CA*s are then used to calculate the measures.

*(1) Entropy Model.* The entropy model determines the amount of movement of a person. We construct a heatmap for each candidate such as shown in Fig. 5. The amount of bins can be specified by the operator. The features of the same person in a *CA* will be mapped to the heatmap based on their position in the camera view. Then we calculate the probability distribution of the bins ($p_{b_i}$) on the heatmap: each bin gets a value indicating the percentage of total features of the *CA* that are in that bin. With the probability distribution we can calculate the entropy over the heatmap ($he_j$). Finally, we weigh the entropy value of each heatmap by their percentage contribution to the total duration of all their *CA*s and then add them all together into the weighted entropy score (*WES*).

$$he_j = -\sum_{i=1}^{n} \left(p_{b_i} \times \log(p_{b_i})\right),\ where\ p_{b_i} = \frac{count(b_i)}{count(CA_j)} \tag{1}$$

$$WES = \sum_{j=1}^{m} \left(he_j \times \frac{Duration(CA_j)}{\sum_{j=1}(Duration(CA_j))}\right) \tag{2}$$

*(2) ReAppearance.* We define reappearing behavior as a situation where someone leaves a camera view and re-enters it again. To model this, we calculate the ReAppearance Score (*RAS*), which is an indicator of how often a person leaves and re-enters the same camera view. We count the amount of *CA*s for each of the cameras a person appears on. The amount of appearance ($a_i$) is used as an exponent in an exponential function ($2^{a_i-1}$) to determine the ReAppearance on that camera. Finally, the *RAS* is calculated by adding the reappearance values of all the cameras together, divided by the amount of cameras that the person appeared on.

$$RAS = \sum_{i=1}^{n} \left(2^{a_i-1}/count(CAMERA)\right) \tag{3}$$

*(3) Duration.* Duration is a general indicator for loitering. The longer someone is in the area, the more likely they are loitering. We calculate the Duration Score (*DS*) of a person by adding the duration of all *CA*s. The duration of a *CA* is the time difference between the first and last timestamp in the *CA*. In the case of *CA*s with overlapping duration intervals, the duration is not added multiple times. As the duration of appearance for each person can vary wildly, we scale down its impact by taking the logarithm over the final duration.

$$DS = \log\left(\sum_{j=1}(Duration(CA_j)) + 1\right) \tag{4}$$

*(4) Loitering Score.* Finally, to create a ranking measure, we combine the three measures into one Loitering Score (*LS*). Different surveilled scenes can have different loitering behaviors that should be distinguished. Therefore we introduce the weights $\alpha$, $\beta$ and $\gamma$. With these the operator can tweak the impact of each on the rank of loiterer candidates. We then rank the people appearing in surveilled scenes in descending order of *LS* to create a list of loitering candidates.

$$LS = \alpha \times RAS + \beta \times WES + \gamma \times DS \tag{5}$$

## 2.2 Implementation

VisLoiter$^+$ is implemented as a Browser/Server (B/S) system according to Fig. 2. The client side frontend can be opened on the browser. From here, the user can send requests to the server, after which the results will be displayed. The GUI is constructed using JavaScript, with the D3 [8] and C3 [11] libraries. The server side backend listens for requests. The loitering analysis and search request listener are implemented in C++ and run on a Ubuntu 16.04 machine.

After receiving a search request, the loitering analysis system will go over each person's the metadata and calculate the measures for them. After that results are ordered and a list of top candidates, with summary information for each, will be sent to the client in JSON format. The summary information consists of a face image, scores for the different measures and all *CA*s for that person. In addition, the loitering analysis stores the intermediate results of the entropy calculations for possible heatmap or entropy detail request.

For the lightweight video playback, a playback request for the selected *CA* is sent to the server. The video playback script then adds the bounding box of the feature of the person for each frame of the *CA*. These frames are then combined into a sequence, which is sent to the client where it will be played back. The heatmap overlay is handled by the heatmap script. It draws the heatmap grid over the frames provided by the video playback. It also requests the heatmap contents from the loitering analyzer and constructs a heatmap image to display next to the video playback. It is worth noting that we never attempt to play back the raw surveillance videos due to heavy network traffic. Thus we designed and implemented such a lightweight video playback for high efficiency instead.

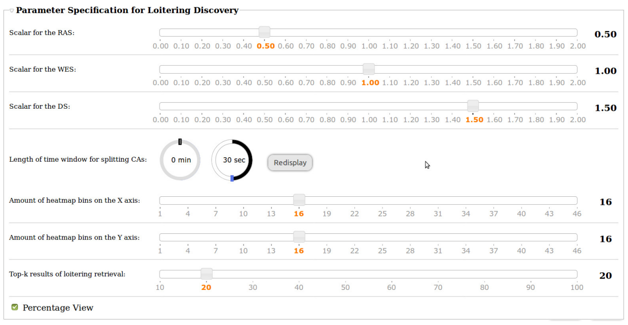
Finally, for the entropy over time view, a stepwise entropy request is sent to the loitering analyzer, which returns the entropy for each intermediate addition to the heatmap, accompanied by timestamp. This data is then used to plot a chart near the video playback box to show the change of entropy over time.

## 2.3 Visualization and Usage

In addition to an improved result ranking, our system features a web-based user-friendly interface. It is optimized for operators to make the correct final decisions, by enabling them to adjust parameters and easily review the list of candidates (see Fig. 4).

The top of Fig. 4 shows a collapsible menu for the *parameter settings*, where the operator can adjust the following system parameters through a series of slider bars (see Fig. 3):

- The weights $\alpha$, $\beta$ and $\gamma$ for each of the loitering measures. Increasing a weight will make that respective measure contribute more heavily to the *LS*. Higher weights result in a higher overall *LS*, but as the *LS* is a comparison measure, only the ratio between the weights will have an effect on the ordering.
- The timewindow $T_{CA}$ that affects the length of the *CA*s. Increasing this will give people a higher "grace period" for leaving and re-entering the camera view, which might be helpful if many people walk along the sides of a camera view.
- The dimensions for the heatmap that is used to calculate the entropy. Using higher dimensions will give more detailed movement information.
- The top $k$ candidates to display. If the operator wants more results, they can increase this value at the cost of a longer processing time.


Parameter Specification for Loitering Discovery

| | |
|---|---|
| Scalar for the RAS: | 0.50 |
| Scalar for the WES: | 1.00 |
| Scalar for the DS: | 1.50 |
| Length of time window for splitting CAs: | 0 min 30 sec |
| Amount of heatmap bins on the X axis: | 16 |
| Amount of heatmap bins on the Y axis: | 16 |
| Top-k results of loitering retrieval: | 20 |

Figure 3: The panel of parameter settings on VisLoiter$^+$.

In addition, there is a toggle for the timescale view. By default, the Appearance Graph shows all the *CA*s in sequence, so their relative length can be easily determined. By toggling this option, the *CA*s will be displayed on a linear timescale. This gives the operator a better overview on when a candidate showed up on the cameras. While these parameters already have default values that give good results, they can still be changed if the operator deems it necessary.

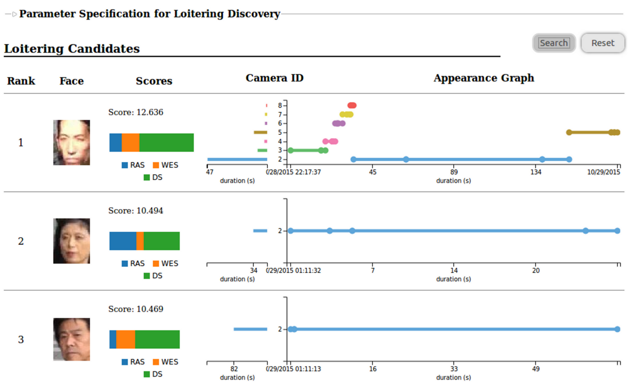


Figure 4: Candidate list overview of VisLoiter$^+$.

Pressing the *search button* will send a search request to the server. After the candidate list is determined, they are sent back to the client and will be displayed in the *Candidate List* below. It gives a detailed overview of the top loitering candidates, with the most likely loiterer at the top. Each *Loiterer List Row* contains the loitering overview of a candidate, each in turn providing some valuable information about that person. They are the *Rank*, *Face Image*, *Score Bar*, *Camera Duration Histogram* and *Appearance Graph*.

The operator can quickly go over the face images in case they are looking for a specific person. Looking at the score bar, the operator can immediately see why the candidate has a high *LS* as the respective measure will have the largest bar segment, such as a high reappearance in the same area (*RAS*), moving irregularly (*WES*) or being present in the area significantly longer than other people (*DS*). Hovering over each of the score bar segments will display the respective measure's value.

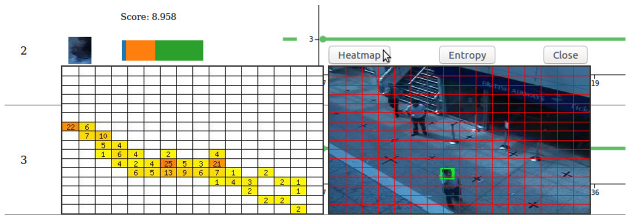


**Figure 5: The heatmap view of the video window. It draws the heatmap grid over the video frame and displays the heatmap for the respective Camera Appearance (*CA*) of a person.**

Next is the duration histogram, which shows the total duration of appearance on each of the cameras. Hovering over each line shows the respective camera's total duration and amount of *CA*s. This shows the operator the amount of cameras the candidate has appeared on, as well as which cameras will be most interesting to investigate further. Note that the duration histogram and appearance graph color of the lines is consistent for each camera. This makes it easy to recognize what *CA* belongs to which camera, and allows a quick comparison between candidates.

Finally, the appearance graph shows all the *CA*s for that candidate. *CA*s are sorted on time and *CA*s with the same camera will be on the same y-level. The appearance graph has two uses. First, it gives the operator a quick overview about the transition of a candidate between cameras and quickly shows if a pedestrian repeatedly entered and exited a camera view. Hovering over a segment will provide the time when the person entered that camera, as well as for how long they were in its view before they left. Secondly, the operator can click on one of the segments to open a video playback window (in Fig. 5 and Fig. 6), which shows the video footage for the respective *CA* of a person.

With these features, operators can further investigate candidates they consider suspicious and then make a final, reliable verdict considering their own expert knowledge.

In addition to the regular footage, there is also a heatmap overlay and entropy over time view, which can be activated by pressing the respective buttons at the top of the video playback window.

The heatmap overlay shows the heatmap grid over the video window, with the dimensions that were provided in the parameters. Next to it, it displays the filled heatmap of that person. This overlay can be used to get a quick idea of how the person moved across the camera view. The heatmap is color coded: the color scale changes from yellow to red, depending on the amount of features in the bin. An example can be seen in Fig. 5. The darker orange spots indicate "walk-stop-walk-stop" behavior.

Finally, the entropy over time view shows a graph with the change of entropy score for a person plotted over time to get an idea about their movement behavior while they were in the area. If the entropy goes up, the person is moving. If it goes down, the person is standing still, and if it stays roughly equal, the person moves in a constrained area. Fig. 6 shows an example of a "walk-stop-change direction" pattern, which can be identified by an upwards trend where the entropy is repeatedly going up and down over time. After the entropy peak, the person stands still, causing the entropy to gradually decrease. Finally, the person goes back the same way they came, causing the entropy to stagnate.

## 3 DEMO

To highlight our improvements compared to the previous approach, we use the AntiLoiter dataset[1] [5, 7] for onsite live demo. We show

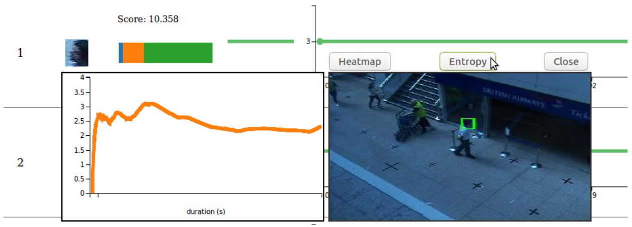


**Figure 6: The entropy over time view. Increasing entropy indicates someone moving, while decreasing entropy indicates someone standing still.**

that the ordering of the candidates is different, due to the new measure being used. In addition we demonstrate the improved overview of the appearances of each candidate. Next, to highlight the effectiveness of the entropy model approach and to show that our system can be used with different feature extraction methods, we use Scenarios 1 and 3 of the PETS dataset[2] as well for live demo. With this part of the demo we illustrate the differences in heatmaps between walking, waiting and loitering people. We also show what the effect of each of these behaviors has on the entropy that is calculated over time. Finally, we demonstrate the effects of adjusting the parameters, by showing the candidate lists based on the parameters inputted.

## 4 CONCLUSION

We created an improved loiter retrieval system called VisLoiter$^+$ and highlighted its improvements compared to previous versions. Our new system can reduce the number of missed candidates by considering a person's amount of movement and reappearances, and provides additional information that is helpful for operators in their final judgement.

## REFERENCES

[1] Héctor F. Gómez A., Rafael Martínez-Tomás, Susana Arias Tapia, Antonio Fernández-Caballero, Sylvie Ratté, Alexandra González Eras, and Patricia Ludeña González. 2015. Identification of Loitering Human Behaviour in Video Surveillance Environments. In *Proc. of IWINAC*. 516–525.

[2] N. D. Bird, O. Masoud, P. Papanikolopoulos, and A. Isaacs. 2005. Detection of loitering individuals in public transportation areas. *IEEE Tran. on Intelligent Transportation Systems* 6(2) (2005), 167–177.

[3] Helen M Hodgetts, François Vachon, Cindy Chamberland, and Sébastien Tremblay. 2017. See no evil: Cognitive challenges of security surveillance and monitoring. *Journal of Applied Research in Memory and Cognition* 6, 3 (2017), 230–243.

[4] Jong-Gook Ko and Jang-Hee Yoo. 2013. Rectified Trajectory Analysis Based Abnormal Loitering Detection for Video Surveillance. In *Proc. of AIMS*. 289–293.

[5] 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 on Multimedia Conference*. 675–679.

[6] Jianquan Liu, Shoji Nishimura, and Takuya Araki. 2016. VisLoiter: a system to visualize loiterers discovered from surveillance videos. In *ACM SIGGRAPH 2016 Posters*. 47.

[7] 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.

[8] Glenn J Myatt and Wayne P Johnson. 2011. *Making Sense of Data III: A practical guide to designing interactive data visualizations*. Vol. 3. John Wiley & Sons.

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

[10] 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.

[11] Masayuki Tanaka. 2013. D3-based Reusable Chart Library. c3js.org. (2013).

[12] Thi Thi Zin, Pyke Tin, Takashi Toriu, and Hiromitsu Hama. 2010. A Markov Random Walk Model for Loitering People Detection. In *Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*. 680–683.

[1]https://github.com/ryukenzen/antiloiter/

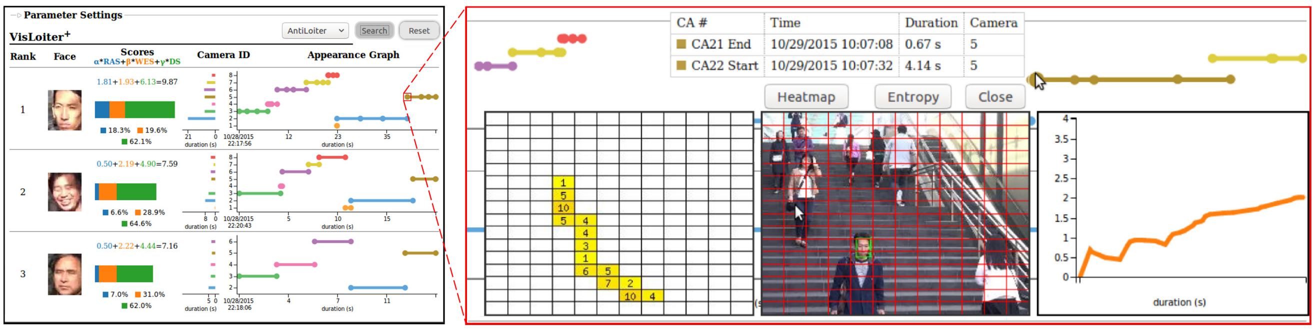
[2]http://www.cvg.reading.ac.uk/PETS2007/data.html

# Chapter 3

# Poster Paper (published at VC 2018)

# Loiterer Retrieval and Visualization using Entropy Model

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



| CA # | Time | Duration | Camera |
|---|---|---|---|
| CA21 End | 10/29/2015 10:07:08 | 0.67 s | 5 |
| CA22 Start | 10/29/2015 10:07:32 | 4.14 s | 5 |

**Figure 1: The system overview and interface of VisLoiter$^+$ [2].**

## 1 OVERVIEW

Surveillance cameras can be used to detect loiterers. However, fully automated detection is challenging. Human operators are very effective at detecting suspicious behavior, but have trouble with focusing on many cameras at the same time. We present a method [1] that analyzes loitering behavior across multiple cameras. It then provides a list of loiterer candidates based on three different loitering characteristics. In addition, we present our system VisLoiter$^+$ [2], which provides the operator with several information channels on these candidates. With this the human operator can easily determine who the actual loiterers are.

### 1.1 Measures

To detect loitering behavior, we use three measures. [1]. The first measure is position entropy. This shows how much a person is moving. The persons movements are mapped to a heatmap. This heatmap shows where the person has been on that camera. It serves as an abstraction of someone's trajectory. Walking people are scored highly, while those that are standing still are scored lowly with this measure. This is used to differentiate loiterers from people who are waiting. The second measure is reappearance. It measures how often someone leaves and re-enters an area. This measure is used to detect people loitering around a larger area. The last measure is duration. People appearing in an area for a long time are more likely to be loitering. These measures are then combined into a loitering score: a higher score indicates a higher likeliness of loitering. With these scores we create a list of candidates ordered by loitering score.

### 1.2 Interface

The loiterer retrieval system VisLoiter$^+$ [2] displays this candidate list and provides additional information for the the operator. An overview of the interface is shown in Fig. 1. The left part is the list of candidates ordered by score. It shows a representative image for each candidate, as well as the portion of the scores. Next to that the camera duration histogram is displayed. This provides a quick overview on which camera a person appeared on, and for how long they were there. After the operator has found a candidate of interest, the appearance graph to the right will provide more detailed information. It shows all the appearances for that person on each camera in chronological order. Each appearances can be clicked to start a video playback. In addition, the appearance heatmap can be displayed. With it the operator can get a quick glimpse of the person's path through the area. Finally the entropy over time is plotted, serving as an indicator of that person's movement. Increasing entropy indicates someone is moving into a part of the area they have not visited before. The entropy decreases if the person is standing still. If the entropy stagnates, the person is moving, but not moving into new areas, such as when someone is pacing. Alternating increases and decreases in entropy with an upward trend are an indication of walk-stop-walk-stop loitering behavior.

## 2 EXTENSIONS

We used the above approach to retrieve loiterers. We aim to expand our model to cover more abnormal behaviors, but it only calculates the entropy over position. By analyzing different behaviors, we found that direction and speed can be used to analyze a variety of abnormal behaviors. Therefore, to expand upon this model, we will also look at the irregularity of direction and speed. These two combined with position form the irregularity measure. We will also provide new visualizations for these measures. For the direction, the heatmap view will color each bin in accordance with its direction. For the speed, a blue-red gradient will be used. Large variety in color shows irregularity in the direction and speed, therefore resulting in a high irregularity measure. We would like to present our retrieval system and discuss the existing work, as well as the extension possibilities with the audience during the on-site poster session.

## REFERENCES

[1] Maguell L. T. L. Sandifort, Jianquan Liu, Shoji Nishimura, and Wolfgang Hürst. 2018. An Entropy Model for Loiterer Retrieval across Multiple Surveillance Cameras. In *Proc. of ACM ICMR (full paper)*.

[2] Maguell L. T. L. Sandifort, Jianquan Liu, Shoji Nishimura, and Wolfgang Hürst. 2018. VisLoiter$^+$: An Entropy Model-Based Loiterer Retrieval System with User-Friendly Interfaces. In *Proc. of ACM ICMR (demo paper)*.