Privacy-Preserving CCTV Analytics for Cyber-Physical Threat Intelligence

Jürgen Neises\textsuperscript{1}, Adrien Besse\textsuperscript{2}, and Jean-Baptiste Rouquier\textsuperscript{2}

\textsuperscript{1} Fujitsu Technology Solutions GmbH, Düsseldorf, Germany
juergen.neises@fujitsu.com
\textsuperscript{2} Fujitsu Technology Solutions SAS, Paris, France
{adrien.besse,jeanbaptise.rouquier}@fujitsu.com

Abstract. This paper describes the FUJITSU CCTV Analytics service (FCAS), a privacy-preserving CCTV surveillance module using AI based video analytics for cyber-physical threat intelligence. The design and architecture of a privacy preserving and thus GDPR friendly way to improve security by automatically analysing video feeds and sending events that can be interpreted as a threat to further analytics is illustrated. The system has been applied to several scenarios like data centre and at ATMs. The developments can also be applied to general public safety requests and may be utilized coping with the COVID-19 impacts. Finally, the solution shall be adapted to edge computing. First steps illustrate the capabilities of small form factor systems.

Keywords: CCTV · Cyber-physical-threat-intelligence · Privacy · Human pose detection

1 Introduction

This document is about physical threat intelligence provided by the FUJITSU CCTV Analytics service (FCAS). Video surveillance is by essence dedicated to monitor assets and interactions with them within a camera scene. Today most security systems are not designed for proactive use but for forensics when a breach has already happened. In our case, the goal is a privacy preserving and thus GDPR friendly way to improve security by interpreting video feeds and sending events that can be interpreted as a threat to further analytics. FCAS will not store any biometric data. Moreover, FCAS is by design agnostic of the security or business use cases it is supposed to support pushing events. The goal is to be able to detect bodies and body parts (e.g. heads, hands, etc.) and capture their interactions with each other and with physical objects (e.g. ATMs or racks). The observations generate events that shall be pushed to data collection for further analytics and correlation.

FCAS provides a set of AI models, able to detect objects out of the box, by using human pose estimation. They are able to detect series of human actions, e.g., opening a door or interacting with an ATM. These interactions result in events indicating potential physical threats. These events can be correlated by AI and analytics identifying physical
threats and indicate to the human operator where to focus his attention or automatically initiate mitigation actions. This way, CCTV systems can be leveraged in order to improve security.

The FCAS CCTV probe relies on one or more CCTV cameras watching a scene, i.e. the complete camera picture. Moreover, relevant static areas – so-called zones – can be easily defined by drawing a polygon around them, thus defining a separated detection area. Then the FCAS can track interactions of dynamic objects within those zones.

FCAS combines the detection of objects, e.g. body parts, and the detection of body poses for analysing the CCTV images. Based on the results FCAS generates action related events. Some models are already available for those tasks, but they often need to be retrained for the specific purpose. Such existing technologies for object detection and pose analysis are combined and retrained to process the video feed in real-time using CPUs or preferably GPUs as in FCAS and to generate events for further security analysis.

1.1 Object Detection

Object detection aims to detect all instances of objects from a known class, such as people, or body parts as heads in an image. Typically, only a small number of instances of the object are present in the image, but they can occur at a large number of possible locations and scales that need to somehow be explored.

Each detection is reported with some form of pose information. This could be as simple as the location of the object, a location and scale, or the extent of the object defined in terms of a bounding box. Providing even more information, a face detector might compute the locations of the eyes, nose and mouth, in addition to the bounding box of the face. The pose could also be defined by a three-dimensional transformation specifying the location of the object relative to the camera.

Object detection systems construct a model for an object class, e.g. heads, from a set of training examples. In the case of a fixed rigid object, only one example may be needed, but more generally, multiple training examples are necessary to capture certain aspects of class variability.

We therefore focus on typical security elements that appear in common video surveillance:

- **Counting People**: The probe sends timestamped events when someone enters or exits the area, allowing getting the number of people in each area.
- **Discriminating Adults from Children**: This element is not implemented in the probe yet but could be added in later versions of FCAS. Most predictions on age are based on face image analysis and require a sufficient level of details for the model to give a reliable age prediction.
- **Face Detection**: The FCAS uses a pose estimation model to detect and track different face attributes such as the nose, eyes and ears of people (cf. Fig. 1). One of the advantages of using pose estimation models instead of pure face detection model is that it can detect people even when the faces are hidden (somebody wearing a balaclava or somebody facing in the opposite direction) from the presence of other body parts.
• **Weapon Detection:** The current version of FCAS focuses on human body parts positions in the scene. FCAS models detecting particular weapons in the hands of somebody are not assessed yet. Major challenges of this approach are creating the training datasets gathering a suitable amount of weapon annotated images extracted from similar security CCTV camera visual context (images from movies contain plenty of weapons but are quite different from CCTV images).

![Body parts definition used by the pose estimation model. Image source Platte et al. [1]](image-url)

Fig. 1. Body parts definition used by the pose estimation model. Image source Platte et al. [1]

### 1.2 Human Pose Detection

Human Pose Estimation attempts to find the orientation and configuration of human body parts. 2D Human Pose Estimation, or Keypoint Detection, generally refers to localize body parts of humans, e.g. finding the 2D location of the knees, eyes, feet, etc. Body parts, which are typically used are depicted in Fig. 1.

Pose estimation is a difficult problem and an active research subject partly because the human body has 244 degrees of freedom with 230 joints. Although not all movements between joints are needed, a common approach is to model the human body by 10 large parts with 20 degrees of freedom [1]. Algorithms must account for large variability introduced by differences in appearance due to clothing, body shape, size, and hairstyles. Additionally, the results may be ambiguous due to partial occlusions from self-articulation, such as a person’s hand covering their face, or occlusions from external objects. Other issues include varying lighting and camera configurations. In the FINSEC case, we estimate pose from monocular (two-dimensional) images, taken from a normal camera.

Human pose estimation is crucial for video surveillance [2]. Recently, significant progress has been made in solving the human pose estimation problem in unconstrained single images. However, human pose estimation in videos is a relatively new and promising problem and needs much further improvement.

Obviously, single image-based pose estimation methods can be applied to each video frame to get an initial pose estimation as illustrated in Fig. 2. A further refinement through frames can be applied to make the pose estimation consistent and more accurate and reliable.
2 CCTV Analytics Technology Description

Within the global architecture in the FINSEC project, FCAS is a probe working in the edge: the FCAS probe uses local CCTV surveillance for analytics and generating events. Converting videos to events is done locally. The events occurring in the local scene are generated in the edge by the FCAS analytics and pushed to a data layer. Anomaly detection and threat prediction utilize the events stored in the data layer for security analysis.

The CCTV FCAS probe architecture is composed of modules that hierarchically process the information contained in the video stream. Figure 3 illustrates the processing flow by the probe from the camera video stream to the events sent to the data layer of the FINSEC platform.

The main modules:

- The **video reader** receives the video frames from the camera
- The **detection** module detects defined body parts and infers their coordinates in those video frames.
- The **tracking** module matches new detections in the last frame to detections in the previous ones, for each body part, thus keeping track of the body part positions. The tracking algorithm uses distances between the current detection and the predicted position (derived from a Kalman filter). (Currently the configuration takes into account frames from the last 12 s). The position of a human body is taken as the position of one key body part (cf. Fig. 4) which in our case is the ‘Neck’ for the pose estimation model or the head for the head detection model.
- The **event generation** module looks for tracking results and generates events about body parts entering or exiting configured areas in an open STIX related format to a security analytics platform.

The body parts illustrated in Fig. 4 can be detected and tracked by the pose estimation model and the human tracker.
To control the number of emitted events, the body parts effectively taken into account for event generation can be set through the FCAS configuration API. By this configuration, the lists of body parts taken into account at event level can be defined to be used for detection of a suspicious event.

In the current version of the FCAS probe with pose estimation, the ‘Neck’ position is selected as the key body part, representative of the body position. Secondary parts are other tracked body parts, e.g. right and left wrists.

2.1 Detection

The detection module uses a pose estimation model or a head detection model to infer the coordinates of body parts in video frames received from the camera. The detection module is responsible for processing a frame and outputs the positions of body parts. The module detects the position of heads, neck or right and left wrists depending on a parameter indicating which body part to track.
The models to detect human bodies are obtained from a pose estimation library, in this case the NVIDIA library “Real-time pose estimation accelerated with NVIDIA TensorRT” [3]. The library authors follow the approach in references [4] and [5], based on part affinity fields and key point detections, for pose estimation. That library provides:

- a pre-trained model zoo for human pose estimation, resulting from several experiments on the MSCOCO Keypoints Challenge 2017 [6]. The experiments are based on several backbone network architectures such as densenet121, densenet169 [7], resnet18 and resnet50 [8].
- a training script allowing to fine-tune the pre-trained model on our specific datasets
- a module to compile model, e.g. using the NVIDIA TensorRT library as described above, allowing to optimize model inference on NVIDIA Hardware such as the Jetson platforms. The library authors report throughputs on Jetson Nano and Jetson AGX Xavier respectively of 22 FPS and 251 FPS with a resnet18 network architecture [3]. The range of available backbone architectures in the library zoo allows optimizing performances versus throughput and latency of the finally deployed model.

The model for head detection uses a MobileNet [9] deep neural network architecture trained on two datasets, namely the Open images dataset [9–11] and the Brainwash dataset [12].

2.2 Tracking

The tracking module is responsible for initializing and updating the following information:

- The object id (here an object is a human body)
- The positions \((x, y)\) in pixels of its body parts
- The area names in which the body parts are present
- The timestamp of entry in the area
- The current timestamp

The tracking algorithm applied in FCAS is based on the SORT algorithm [13]. This algorithm is effective and uses position extrapolation. The first approach of this algorithm is that objects do not move a lot between frames: if the FCAS system detects an object in one frame, 100 ms seconds later it cannot be far from where it was before. In addition, it computes a direction and predicts the new location in the current frame. The FCAS probe searches for the object next to where it is predicted to be. Predicting the path of the object allows following the object even if someone else temporarily occludes it, which means that only the occluding person is detected. Note that this algorithm preserves privacy as the FCAS tracks objects by predicting their future position and not by recognizing them.

In order to efficiently address the tracking of human bodies through video, we have developed a human model using individual trackers for each body parts and combining the information of all the body part trackers to keep track of the human body. The building blocks are trackers based on Kalman filters [14] for each body part position.
prediction. The matching between tracker predictions and detection is performed following the Hungarian algorithm [15]. Open source implementation of those are Python packages filterpy.kalman and scipy.optimize.linear_sum_assignment. From the building block consisting in one tracker, we have developed a global human tracker combining a set of body part trackers. Each human model tracker has a unique ID. This way pose estimation outputs based on various body parts make human model tracking more robust than a tracker based on a single human detection, as relies on more information than just on a unique bounding box detection.

The Human model tracker has internal states such as centre positions and velocities for each part of the body. The event generator to monitor human bodies in the video can access the human model internal state. In particular, left and right ankle positions can be used to map the human positions on a floor map of the scene.

2.3 Event Generation

The FCAS collects and processes the video stream of a camera and stops if the video stream ends. The event generator module is responsible for sending events generated from the tracker inputs in a STIX like format. We outline in this section the various events that can be generated by the FCAS.

**New Agent Events**

When a new body is detected in the camera field or scene for more than a defined period (e.g. 2 s), the FCAS creates a new agent instance with new id and sends an ENTER event.

![Fig. 5. Enter/Exit Event](image)

When a body disappears from the camera field of view, the FCAS generates and sends an EXIT event for this from the last area it appeared in, and cleans the information of this body.

**Enter/Exit Events**

When a body part enters a scene area, e.g. Zone A for the time, FCAS sends an “entering area ‘Zone A’” event. Later, when a body part (e.g. LWrist as illustrated in Fig. 5),
exits from the ‘Zone A’ area and enters into the ‘Zone B’ an area, FCAS sends the corresponding events “ENTER (Zone B)” and “EXIT (Zone A)”.

**Velocity Related Events**
These events are related to the velocity of bodies moving through the scene. The neck is the default reference point defining the body and thus its velocity. The events are illustrated in Fig. 6.

![Fig. 6. High Velocity and Slow Down](image)

The configurable value body_speed_threshold defines the threshold above which to trigger **high velocity** events.

If the velocity of a reference point, e.g. 10 cm per second is shrinking an event indicating **slow down** of the agent is issued.

**Proximity Related Events**
These events consider the proximity of two bodies. The value parts_body_proximity specifies a list of body parts used to determine if two human bodies are closer than a given threshold (cf. Fig. 7). The values defined in body_radius_proximity set the corresponding thresholds defining body proximity.

![Fig. 7. Approaching and Distancing](image)
Two events are related to body proximity:

- APPROACHING when the distance is below the set threshold and shrinking
- DISTANCING when the distance is above the defined threshold and increasing

**Trajectory Related Events**
Loitering is a major security event and is worth monitoring. Thus, the body trajectories are tracked on a floor map within the scene. Additionally, their length is compared with length_floormap_trajectory and length_trajectory_threshold. When the length of the trajectory in the floor map exceeds the defined threshold (cf. Fig. 8), the TRAJECTORY event is triggered. In combination with the time an agent is in the scene this event indicates loitering.

![Fig. 8. Trajectory Event](image)

### 3 Application to Common Physical Security Scenarios

In this section, the impact of the FCAS are outlined related to two common physical security scenarios. These comprise physical security in the data centre and at ATMs.

#### 3.1 Scenario Data Centre

A well known security scenario in the data centre is the protection of access to racks, containing secured servers. By a common security policy, a secure rack has to be opened
by two people, each one using an authorized key. All the other configurations (one person using two keys, two people coming sequentially, etc.) are unauthorized in this case.

A camera on top of the rack may survey the scene. This way it can be detected, when someone enters the area in front of a rack and if the person uses the keyhole with one of their wrists. The FCAS probe allows tracking interactions of humans with key holes, counting the number of people, and tracking their positions in different areas. Events indicating the activities and potential protocol violations are submitted to a security platform and may be used by anomaly detection to trigger security actions at the data centre scale. In case, an anomaly triggers alarms and pushes them to the data centre operations.

3.2 Scenario ATMs

ATMs continue to be a profitable target for criminals. While some rely on physically destructive methods with metal cutting tools or explosives (including injecting gas), others choose malware infections. Criminals use different tools and techniques (physical, logical or combined) to access ATMs as a “black box” and bypass all security controls forcing the cash-out of all its money, cut communication with the ATM host in order to avoid any interference from operational personnel monitoring the smooth operation of the ATMs.

In this scenario, two camera feeds shall be utilized in the FCAS solution. Typically, a dome camera is monitoring the scene at an ATM. Additionally, a portrait camera on top of the ATM can be used to ensure detection of a person in the zones close to the ATM using head detection (cf. Fig. 9). This way, the precise position of the person related to the ATM left, centre, right can be detected.

![Fig. 9. CCTV Analytics in the ATM Pilot](image)

In addition, FCAS can monitor the number of people, the time spent in front of the ATM on in the monitored area and if they arrived in the same time in front of the ATM.

Among others, there are five well-known use cases in this scenario and several CCTV related events can be used for combined cyber-physical threat intelligence:

- **Attack to a person at the ATM.** To indicate such an attack, two or more persons need to be in the scene and they need to get close to each other.
Privacy-Preserving CCTV Analytics for Cyber-Physical Threat Intelligence

- **Attack to an ATM.** To attack an ATM, one or more people need to be in the ATM zone.
- **Loitering.** A person staying in the scene for longer than a specific time indicates this. In addition, a trajectory longer than a defined length may add confidence to this.
- **Introduction of malware to the ATM.** From CCTV point of view, this is analogous to Attack to an ATM since the ATM needs to be manipulated.
- **Jackpotting.** This also relates to the second use case “Attack to an ATM” considering CCTV events since the ATM needs to be manipulated.

Moreover, events of cyber and physical probes can be correlated by an anomaly detection and when attacks are detected, alarms can be issued or other mitigating actions can be initiated.

### 3.3 Application to Other Segments

The FCAS development is based on an early very general Proof-of-Concept in the retail segment. Beyond the described scenarios the ongoing FCAS development has been considered for retail based use cases.. Among those, use cases counting people entering shops, monitoring how long people stay in front of items, and interactions with sellers.

With the spring 2020 COVID-19 crisis, the requests for counting people inside an area and monitoring their distances generated further application scenarios for this technology. Especially limited access to shops can be controlled counting people. Moreover, utilizing floor maps of critical areas, e.g. at the cashiers desk, distances between people can be monitored. The latter might be applied also to critical public spaces.

### 4 Deployment – from Server to Edge Computing

During the FCAS development, a trend towards application of AI in the edge has been observed. Therefore, a deployment in the edge has been explored.

The development starting point was applying existing technology based on a system with a powerful GPU, e.g. an X64 system and at least a GPU as powerful as an NVIDIA 1080GTX. This limits the utilization of the FCAS due to the system requirements and related cost. Therefore, the system was not easily deployed in the edge.

This is the reason to evaluate small form factor systems, in this case the NVIDIA Jetson platform, namely the Jetson AGX Xavier, which is expected to provide the required performance at lower cost than a typical server-based GPU and at a small form factor, as a first step towards edge computing or even embedded application, e.g. inside an ATM.

With this process applying a further specialized computer architecture, i.e. ARM in addition to x64, we faced common issues in the dynamic development of new hardware and open source software as depicted in Fig. 10.

Even between servers with the same operating system (Ubuntu 18.04) the libraries and versions of the underlying software change. This implies problems in finding the right Python packages version in repositories for the specific ARM architecture. Moreover, conflicts occurred as some parts required different versions of a given Python library.
The precise list of packages and versions would be fastidious to list. The main libraries that faced such issues are scikit-learn and scipy (both needed for the tracking algorithm).

To speed up deep learning inference as compared to TensorFlow, we moved to the TensorRT framework published by NVIDIA. It provides optimizations for GPU-based platforms. This requires usage of a distinct, dedicated open source code repository for pose estimation instead of the original one (https://github.com/NVIDIA-AI-IOT/trt_pose). The latter is written in pytorch and needs to be compiled to be optimized on each hardware (Jetson GPU or common GPU), while the former one used an open source TensorFlow implementation of pose estimation. However, this replacement may be advantageous for both hardware environments (edge and server) as it the optimisation through TensorRT may be used for underlying hardware compatible with the library. This way an optimization can be achieved for both computing environments by this pose estimation implementation.

5 Conclusion

This paper shows a privacy-preserving CCTV surveillance module using AI based video analytics for cyber-physical threat intelligence. It is based on common technologies for detection of body parts and human pose detection. There is an ongoing development to move the solution towards and edge based intelligent sensor of potential physical threats.

Acknowledgement. A substantial part of this work has been carried out in the scope of the HORIZON 2020 FINSEC project, which is funded by the European Commission in the scope of its HORIZON 2020 programme (contract number 786727). The authors gratefully acknowledge the contributions of the funding agency and of all the project partners.

Besides the authors, Sebastien Iooss, Moïse Valvassori, Ivaylo Petkanchin, Thierry Lefort, Arthur Cahu and Thomas Walloschke were involved in the work of Fujitsu side.
References

1. Platte, B., et al.: Person tracking and statistical representation of person movements in surveillance areas. Int. J. Des. Anal. Tools Integrated Circ. Syst. 7(1), 10 (2018)
2. Zhang, D., Shah, M.: Human pose estimation in videos. In: 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, pp. 2012–2020 (2015). https://doi.org/10.1109/iccv.2015.233
3. NVIDIA: “Real-time pose estimation accelerated with NVIDIA TensorRT”. https://github.com/NVIDIA-AI-IOT/trt_pose
4. Zhe, C.G.: OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields, arxiv, 1812.08008 (2018)
5. Xiao, B., Haiping, W., Yichen, W.: Simple baselines for human pose estimation and tracking. In: Proceedings of the European Conference on Computer Vision (ECCV) (2018)
6. MSCOCO Keypoints Challenge (2017). https://cocodataset.org/#keypoints-2017
7. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2261–2269 (2017). https://doi.org/10.1109/cvpr.2017.243
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770–778 (2016). https://doi.org/10.1109/cvpr.2016.90
9. Howard, A.G., et. al.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, arXiv e-prints, arXiv:1704.04861. April 2017
10. Krasin, I., et. al.: OpenImages: A public dataset for large-scale multi-label and multi-class image classification (2017)
11. Kuznetsova, A.: The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale, arXiv, arXiv:1811.00982 (2018)
12. Brainwash data set. https://megapixels.cc/brainwash/
13. SORT: SIMPLE ONLINE AND REALTIME TRACKING. https://arxiv.org/pdf/1602.00763.pdf
14. Kalman, R.: A new approach to linear filtering and prediction problems, J. Basic Eng. 82, 35–45 (1960)
15. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Research Logistics Quarterly 2, 83–97 (1955)