# Privacy-Preserving Moving Object Detection

Jonathon Ackers

## Description

The inspiration for this project is smart home and security companies such as Ring. These companies offer their customers the ability to secure their homes by selling a variety of surveillance cameras for both internal and external use. Moreover, as well as traditional surveillance, companies are now utilising the latest machine learning technologies to analyse video streams in order to interpret what is being recorded, and respond accordingly. For example, if a smart doorbell recognises who is arriving at the front door, it could automatically unlock and allow them to enter. However, while there are many benefits to this - and these systems often makes peoples' lives easier, there are downsides to security cameras getting smarter.

In its current form, machine learning inference requires more hardware power than can be included in the relatively small devices. Consequently, video data must be transferred to servers where the analysis can be performed. During the transfer this data will be encrypted for security and privacy, but when it arrives on the cloud, it must then be decrypted so that the machine learning models can be applied. This means that the companies providing these services have access to raw video streams of peoples' homes. Understandably, this has become a privacy concern for many people whom don't trust these companies not to be malicious with this data. For example, service providers could use this data to monitor their customers' property and guests, then sell this to other companies. The incentive for the organisations to implement a system like this is to provide assurances to their customers that their privacy will be protected, so should encourage more people to use their products.

Therefore, the goal of my project is to create an application which can perform machine learning inference on encrypted data, rather than raw video, in order to preserve the privacy of users. The application will emulate the services provided by companies like Ring, in order to demonstrate what could be possible in real-world industry. For simplicity, and to ensure the solution is as robust as possible, I will focus on just moving-object detection. Initially, I plan to use a standard homomorphic encryption library, such as SEAL or the Paillier cryptosystem, for the encryption aspect of the project. Homomorphic encryption is a method of encryption design to permit computations to be performed on the encrypted data without first decrypting it. For example, the numbers 3 and 4 could be encrypted, then the encrypted values could be multiplied together, and decrypting the result would give 12. This will allow me to focus on building and optimising the machine learning algorithms which will perform the moving object detection. The result of this stage of the project will allow users to upload a video file, and receive a video highlighting any moving objects. Part of this phase may require training machine learning models to perform inference. Training can be very slow, so it is important that I leave enough time during my project to ensure the model has learned enough to produce sufficiently accuracy results.

Once the homomorphic encryption solution has been successfully implemented, I will attempt to speed up the running time of the application. To do this, I will try to optimise the encryption algorithm, in an attempt to create a scheme which is much less computationally expensive to perform inference on. As part of this, I will draw from some published works. However, in order to increase the speed of the application, some trade-offs will have to be made. Therefore, during the final stage of the project, I will compare the faster encryption scheme to the original homomorphic scheme; allowing me to gain an understanding of them relative to each other, using metrics like running time, accuracy, memory usage, and practicalities of engineering, as well as evaluating the security of the cryptographic algorithms, possibly by simulating attacks on the system.

## Timetable

| **Package** | **Start** | **End** | **Description** |
|---|---|---|---|
| 1 | 14/10/2021 | 27/10/2021 | *Preparatory work*. E.g. experiment with hardware limitations to determine if extra resources required and if application can be entirely client-side, or if a client-server model will need to be developed. |
| 2 | 28/10/2021 | 10/11/2021 | Begin creating simple video handling application. Milestone: Complete frontend of application. *Introduction to Robotics Assignment 1 Deadline: 01/11/2021* |
| 3 | 11/11/2021 | 24/11/2021 | Implement homomorphic encryption algorithm. *Introduction to Robotics Assignment 2 Deadline: 22/11/2021* |
| 4 | 25/11/2021 | 08/12/2021 | Begin implementing inference algorithm. |
| 5 | 09/12/2021 | 22/12/2021 | Finish inference algorithm. Milestone: Full stack implemented. |
| 6 | 23/12/2021 | 05/01/2022 | *Buffer.* |
| 7 | 06/01/2022 | 19/01/2022 | Begin implementing second encryption algorithm. |
| 8 | 20/01/2022 | 02/02/2022 | Finish second encryption algorithm. Milestone: Implementation complete. |
| 9 | 03/02/2022 | 16/02/2022 | Write disseration: Introduction and Implementation chapters, and share with supervisors. *Cybercrime 1 Deadline: 04/02/2022* |
| 10 | 17/02/2022 | 02/03/2022 | Evaluate two approaches by recording the accuracy, running time, and memory usage. *Cybercrime 2 Deadline: 18/02/2022* |
| 11 | 03/03/2022 | 16/03/2022 | *Buffer.* *Cybercrime 3 Deadline: 04/03/2022* |
| 12 | 17/03/2022 | 30/03/2022 | Write dissertation: Evaluation of approaches, and share with supervisors. Milestone: First full draft. *Cybercrime 4 Deadline: 18/03/2022* |
| 13 | 31/03/2022 | 13/04/2022 | Write dissertation: Evaluation of project and response to feedback on first draft, and share with supervisors. Milestone: Second full draft. |
| 14 | 14/04/2022 | 27/04/2022 | Write dissertation: Response to feedback on second draft, and share with supervisors. Milestone: Final draft. |

## Special Resources

Running machine learning on inference on video streams requires large amounts of memory. This is because the video must be loaded into memory entirely before inference can occur, rather than simply reading single frames like when images are analysed. Therefore, I will likely require more GPUs, to create a greater memory capacity, in order to meet the VRAM requirements. How much support I require will become clear once I begin implementing prototype systems.

Initially, I plan on using the Moving MNIST dataset to simplify execution of the application. However, towards the end of the project I hope to use videos more similar to real-life surveillance systems. While I should be able to synthesise this data myself, the videos could contain images of other subjects, so I may require approval from the ethics committee.

## Starting Point

I have not completed any work on this project in advance of submitting this proposal. All code will be written from scratch during the time allotted in my timetable. I also have little prior knowledge about homomorphic encryption and moving object detection, so I have included time for research during the preparation stage of my timetable.

The inspiration for this project has come from several papers - which I will be using to assist me during the implementation phase of the project. I have listed all of the papers I am currently planning on using here:

- Real-Time Privacy-Preserving Moving Object Detection in the Cloud by Chu et al.
- Moving Object Detection in the Encrypted Domain by Lin et al.
- Privacy-Preserving Watch List Screening in Video Surveillance System by Sohn et al
- Adaptive background mixture models for real-time tracking by Stauffer et al.
- Object Detection in Encryption-Based Surveillance System by Zeng et al.

## Success Criteria

The project will be deemed successful once the following criteria have been met.

1. The application allows users to provide a video file to the client, which can then homomorphically encrypt the file, and transfer it to the server. The server can send the encrypted data back to the frontend, where it can be decrypted and played to the user.
2. The application implements a statistical model which can detect moving objects on encrypted videos from the Moving MNIST Handwritten-Digits dataset using Gaussian Mixture Models.
3. The accuracy of the moving object detection should be evaluated: comparing inference on both unencrypted, and homomorphically encrypted video, using the same statistical modelling. Metrics such as running time and memory usage will also be compared.

## Extensions

If time allows, I will consider completing the following extensions.

1. I will optimise the encryption scheme used by the application in order to speed up the running time of the application, and hopefully create a real-time solution to moving object detection. If this is successful, I will allow users to live-streamed video from sources like webcams rather than having to upload a pre-recorded video file.
2. I will provide a security analysis of the the bespoke encryption algorithm, in order to evaluate the trade-offs necessary to speed up running time. This could consist of an analysis of the mathematics used during the implementation of the algorithm, or a practical attempt to break the algorithm, such as using penetration testing.
3. I will use machine learning to perform object recognition on the encrypted video, once the moving objects have been segmented. Therefore, as a result of the inference, users will receive a video highlighting moving objects, as well as a description of what these objects are.