# Green light- Red light

## INF-1600 - 14.11.2022

Authors: Sindre Franzen Dalvik and Sivert Bottolfsen

## Abstract

The topic areas are human recognition, and motion detection. The project will create a game using computer vision and deep learning neural network, to decide if the player breaks the rules of the game. The result uses human recognition to decide what part of the video feed should be scanned for movements. This implementation is a working game, but just barely, lacking a proper user interface and graphics. Image recognition and movement-mask are used together to make the game work.

A video demonstration of our implementation:
https://youtu.be/oRQj2ErvTRc

### Keywords

Ai
Deep learning
OpenCV
MOG2
YOLO
Background subtraction algorithm
Motion-tracking
Squid game
Camera

# 1 Introduction

A “red-light green-light” game. A light alternates between green and red. Movement is only allowed when the light is green. If a player moves while the light is red, the player loses. Motion tracking/image recognition will be used to act as the judge. The player starts at a distance of around 8 meters from the computer, and wins if he/she manages to walk all the way to the computer and push the space button.

## 1.1Background and Related work (so called State-of-the-art and motivation)

Object tracking is an application of deep learning. The task of object tracking is to automatically identify objects through a video or live camera. The applications for object

tracking are human-computer interaction, security and surveillance, video communication and compression, augmented reality, traffic control, medical imaging and video editing.

Object tracking can be very challenging due to the amount of data contained in the video. The process can be even more challenging with the possibility of adding object recognition for tracking of targeted objects.

Object recognition will be used for this project. The reason is that the game should only detect the motion of humans. Face recognition will not be used since other body parts also need to be detected for movement.

The method used for tracking objects is a computer vision object detection algorithm called YOLO (You only look twice). YOLO uses a deep learning network consisting of multiple convolutional networks. YOLO was chosen because it is one of the fastest object detection algorithms today. The reason it is called “you only look once” is because it only requires one forward propagation through the neural network to classify and detect bounding boxes for the whole frame. [5]

The object detection will detect many different objects, not just humans. For this implementation, only human detection is needed. But since the model for the network is pre trained, it includes more objects than humans. The network could be trained for only humans, but a lot of good training data would be necessary for us to train the human recognition network. It would have been possible to find and download training data from the internet, but we decided to instead download an already trained data set. The green light, red light game could work for animals too since the model tracks dogs and cats, but only human support is currently added.

YOLO can find specific objects in the image and show boxes around them, but a problem with it is to actually track motion. Therefore the project uses two algorithms for detecting if a person moves. YOLO detects if the object in the frame is a human, and the method used for detecting motion is a background subtraction algorithm which is a widely used technique for generating a foreground mask which is an image containing only pixels belonging to moving objects.

The method will use a probabilistic model called the Gaussian mixture model to model the background pixels where it uses the color of a pixel to determine if the pixel belongs to the background. The weight of the distribution of the pixels is proportional to the amount of time the same color stays on the same pixel. Therefore, the longer a pixel has the same color in

the same position, the higher weight that pixel will have, making it more likely to be classified as part of the foreground. [2]

This object tracking technique is called MOG2 background subtraction. MOG2 is based on two papers by Z. Zivkovic "Improved adaptive Gausian mixture model for background subtraction" in 2004 and "Efficient Adaptive Density Estimation per Image Pixel for the Task of Background Subtraction" in 2006. [3]

The advantage of this method is that there is no need for a background image to initialize the program. Instead, every frame is used for determining the background and updating the foreground mask.

MOG2 was also chosen since it is shown to have better performance compared to other algorithms such as MOG and GMG. MOG2 has a higher precision rate and lower processing time for each frame. [4]

# 1.1 Creators

The MOG motion detection algorithm was retrieved from a youtube video [6] and modified to work with red light green light, mostly by Sivert. The YOLO object detection algorithm was retrieved from a github repository [7] and modified to work with red light green light, mostly by Sindre. Connecting these systems together was done by both of us. Timing of the red light and green light and sounds was done by Sivert.

# 1.2Problem statement, problem and Aim

**Problem:** How can we implement AI to act as a judge in a “red-light green-light” game?

**The ideal implementation** would allow for several players to compete against each other. Also the AI would differentiate between the players, and display an indication that a player has lost if he moves when not allowed.

**Some problems**:
The separation of players for instance, if we were to implement multiplayer. The problem is that once one player moves, and loses, when he walks out of the playing field, the motion detection would still detect him, potentially ruining the game for the other players. While if the game is single-player, The whole game would end when motion from the one player is detected. We therefore decided to fo for a singleplayer game. (potentially implementing multiplayer functionality at some later point). Another problem would be that movement from non-player sources could trigger an accidental game-over. To solve this, we use two computer vision systems in tandem, one which uses object recognition to create a bounding box around humans (the player). And another to look for movement inside that box.

Another problem is the finetuning of the sensitivity of the tracer system. So that some movement is allowed, but not too much. This comes down to fine tuning the variables (which could vary from computer to computer).

**The aim** is to make a single-player game, with a ui and sound from the scene in “squid game”. The system makes a sound and displays on the UI when movement is allowed or not. If movement is detected a gun sound is played, and the UI should inform that the game is over. The game would be won if the player manages to move all the way to the laptop that runs the game, without getting caught moving (while the light is red). We will also implement a timer, like in the series. If the player has not reached the computer before the time is up, the game is lost. To notify the program that the user has reached the goal/the laptop, we will implement an input, for example pressing the spacebar, in which case the player wins.

Though we originally wanted a multiplayer game, with a great looking user interface (GUI), we will prioritize getting the human detection deep learning network and the game to work, then adding a UI and sounds if we have time.

# 2 Implementation and Design

YOLO is used to find the humans in the camera frame. If it finds humans in the camera frame, it will draw a rectangle around the humans. Rectangles are drawn around other objects as well, but are ignored in this project.

The MOG algorithm relies on the boxes found from the YOLO algorithm. The MOG algorithm finds all the moving pixels in the frame. Since it only should detect humans moving, it finds out how many moving pixels there are in the box around a human found from the YOLO algorithm. If the ratio between pixel-density from MOG and the box area is greater than some value, the person in the box will be marked as moving. The reason a ratio of the pixel-density and box area is used instead of only using the total amount of pixels in the box is because the person is able to move away or closer to the camera. The further away a person is to the camera, the less pixels there are in the box to the person, and the closer to the camera a person is, the more pixels there will be. However, the box area will also shrink when moving away from the camera, and expand closer to the camera. Therefore a ratio between the density and area makes the total amount of motion the same regardless of the distance to the camera.

Both the MOG and YOLO algorithm need to use the same camera. This was done by storing a variable for the same camera, and passing it to both of the algorithms, and the systems are connected through the main file.

The game works best when using a nvidia GPU because of the YOLO algorithm. The game can be played using only the CPU, but will be extremely laggy. Since motion detection requires a decent frame rate, the game is pretty much only playable with a high frame rate which is only achieved by using GPU.

# 2 Consequences and Ethics

There are not that many potential risks with our game, as it is just a game. One interesting ethical “problem” could be that the system could potentially have more difficulty tracking people of different ethnic/other groups. But this is not something we have seen happen with our implementation.

# 3 Conclusions and Discussion and summary

We used OpenCV for python for image recognition, and movement detection. We achieved parts of the goals stated in **“the aim”**, but we have not implemented:
-Gun sound
-UI (camera feed is implemented, but no squid game graphics, or game over/win graphics
-A timer for the game (we have however implemented timer between green and red light)

The goal was achieved in the end. YOLO is used to find borders around humans playing the game, and MOG2 is used to detect motion inside these borders. YOLO made use of deep learning with multiple convolutional network layers, and MOG2 used a probability distribution to determine if it is part of the background or foreground.

# 4 Reference(s)

1. Wikipedia contributors. (2022, July 22). Video tracking. In *Wikipedia, The Free Encyclopedia*. Retrieved 19:20, October 10, 2022, from https://en.wikipedia.org/w/index.php?title=Video_tracking&oldid=1099860773
2. Geekering. (2021, Nov 24). BackgroundSubstractorMOG2. Retrieved 21:55, October 10, 2022 fromhttps://www.geekering.com/categories/computer-vision/anilyadav/how-to-detect-motion-using-background-subtraction-algorithms/?utm_source=rss&utm_medium=rss&utm_campaign=how-to-detect-motion-using-background-subtraction-algorithms

3. Stack Overflow. (2019, Sept 23). BackgroundSubstractorMOG2. Retrieved 21:50, October 10, 2022 from

https://stackoverflow.com/questions/33266239/differences-between-mog-mog2-and-gmg

4. A Comparison between Background Modelling Methods for Vehicle Segmentation in Highway Traffic Videos. (2018). L. A. Marcomini, A. L. Cunha. https://arxiv.org/pdf/1810.02835.pdf

5. Yolo — you only look once - towardsdatascience.com. (n.d.). Retrieved November 14, 2022, from https://towardsdatascience.com/yolo-you-only-look-once-3dbdbb608ec4

6. Object Tracking with OpenCV and Python. from https://www.youtube.com/watch?v=O3b8lVF93jU

7. AS-One: A Modular Library for YOLO Object Detection and Object Tracking https://github.com/augmentedstartups/AS-One