Universidad de Alcalá
Escuela Politécnica Superior

Máster Universitario en Ingeniería de Telecomunicación

Trabajo Fin de Máster

Gaze orientation to evaluate object interaction in human functional assessment

Autor: Álvaro Nieva Suárez

Tutor: Marta Marrón Romera

2022

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Trabajo Fin de Máster

Gaze orientation to evaluate object interaction in human functional assessment

Autor: Álvaro Nieva Suárez

Director: Marta Marrón Romera

Tribunal:

Presidente: Juan Carlos García García

Vocal 1: David Fernández Barrero

Vocal 2: Marta Marrón Romera

Calificación: .....

Fecha: .....

Luck is sought, not found

Acknowledgements

A mi familia y amigos por aguantarme todos estos años.

A Marta Marrón por guiarme en el desarrollo de este proyecto, iniciandome en el mundo de la investigación, el cual no sabía ni que existía, ni que podía entrar.

Álvaro Nieva Suárez

Resumen

El objetivo de este proyecto es desarrollar un sistema automatizado para la evaluación objetiva de las limitaciones funcionales, que se enmarca dentro del proyecto de investigación: "Multisensorial analysis of human activity for diagnosis and early detection of functional limitations" (EYEFUL, PID2020-113118RBC31), cuyo nombre en castellano es "Análisis multisensorial de la actividad humana para el diagnóstico y detección precoz de las limitaciones funcionales".

En dicho proyecto de investigación, los OT quieren eliminar la subjetividad del examinador cuando se está evaluando con la técnica AMPS, que es una herramienta utilizada para evaluar las ADL de una persona. Para ello se han seguido los siguientes pasos.

En primer lugar, se añade una introducción para abordar el proyecto. A continuación, en el apartado de estado del arte se comparan diferentes métodos con el fin de elegir la mejor opción para este proyecto. En el siguiente capítulo se explica el flujo de trabajo del sistema, centrándose en la interconexión entre los diferentes bloques utilizados en este proyecto. En el siguiente apartado se recopilan las bases de datos y métricas utilizadas en este TFM, ya que se utilizarán en el apartado de resultados. Finalmente, se añaden los resultados cuantitativos y cualitativos y se exponen algunas conclusiones.

El reto de este proyecto es interconectar todas los bloques utilizados y conseguir un buen rendimiento. Además, será necesario ajustar los diferentes parámetros de los algoritmos utilizados, ya que la configuración por defecto podría no ser la mejor opción.

Keywords Estimacion de la mirada, detecci on de objetos.

Abstract

The purpose of this project is to develop an automated system for the objective evaluation of functional limitations, which is part of the following research project: "Multisensorial analysis of human activity for diagnosis and early detection of functional limitations" (EYEFUL, PID2020- 113118RBC31).

On that research project, the OT wants to eliminate the subjectivity of the evaluator when he or she is evaluating with the AMPS technique, which is a tool used to assess a person's ADL. To achieve that the following steps have been followed.

First, an introduction is added to address the project. Then, on the state of art section different methods are compared with the purpose of choosing the best option for this project. On the next chapter, the system's workflow is explained, focusing on the interconnection between the different blocks used on this project. The following section compiles the used databases and metrics on this TFM, as they will be used in the results section. Finally, quantitative and qualitative results are added and some conclusions are stated.

The challenge of this project is interconnecting all the layers used and achieving a good performance. Also, it will be needed to adjust the different parameters of the used algorithms, as the default configuration might not be the best option.

Keywords: Gaze estimation, object detection.

Contents

List of Figures

List of Tables

List of Acronyms

ADL	Actividades de la vida diaria, Activities of Daily Live.
AI	Artificial Intelligence.
AMPS	Assessment of Motor and Process Skills.
AUC	Area Under the Curve.
BB	Bounding Box.
CNN	Convolutional Neural Network.
DL	Deep Learning.
GT	Ground Truth.
IoU	Intersection over Union.
ML	Machine Learning.
NMS	Non-maximum Suppression.
OT	Occupational Therapist.
POV	Point Of View.
SOTA	State of Art.
SVM	Support Vector Machine.
TFM	Trabajo Fin de Master.
URJC	Universidad Rey Juan Carlos.

Chapter 1

Introduction

The following section explains the reasons and motivation to work on this project.

This Trabajo Fin de Master is related to the following research project: "Multisensorial analysis of human activity for diagnosis and early detection of functional limitations" (EYEFUL, PID2020- 113118RB-C31). The purpose of the project is to develop an automated system for the objective evaluation of functional limitations. This would allow the Occupational Therapist (OT) a tool that can give objective assessments with clinical validity. To achieve our objective, Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning (DL) methods have been used on RGB videos.

This project fits in the "Space and Defense Technologies" specialization of the Telecommunications Master, as it can be used to evaluate person's ability to do a task in possible handicapped circumstances. The system to be developed will help the instructor to compare the professional evaluation with a blind one, in order to validate and defense users' abilities progression in an aseptic context.

1.1 Motivation

In OT, occupations refer to everyday activities (Actividades de la vida diaria, Activities of Daily Live (ADL)) that people do as individuals, in families and with communities to occupy time and bring meaning and purpose to life [12].

In the traditional approach OT, improved occupation was supposed to come with treatment of a patient's underlying physical issues, but they realized the patient improved faster if the therapy was focused on the occupation [13]. Then, they started to evaluate the quality of how the person accomplished those ADL instead of assessing the underlying body functions of the patient. The technique used for this aim was called Assessment of Motor and Process Skills (AMPS).

AMPS is an instrument that allows an objective and standardized assessment of the individual's skills in the performance of ADL, in a natural and relevant environment [14]. It has over 125 chores, with different degrees of difficulty (simple self-care activities to chores with a high complexity and different stages) and can be performed on different environments (at home, supermarkets, parks...).

The main concern of this approach is the subjectivity of the OT, which will influence the output of the evaluation. Besides, the OT must be close to the patient as the assessment is done by watching the person doing the chore and his or her presence could interference on the output of the assessment [15]. These problems can be solved by designing an automatic system that can validate clinical tests of functional limitations and this thesis focuses on developing a part of the proposed system.

This project's goal is to determine the percentage of time a person is focused on a task by estimating the gaze orientation and checking if the patient is focused on the object assigned to the task: the gaze direction is a key element to understand people's intentions, what they are doing and it is needed to evaluate person's abilities [16].

1.2 Objectives of this project

This section includes the scope of this project. On figure 3.1 the system's workflow can be seen, whose components are explained in the following sections.

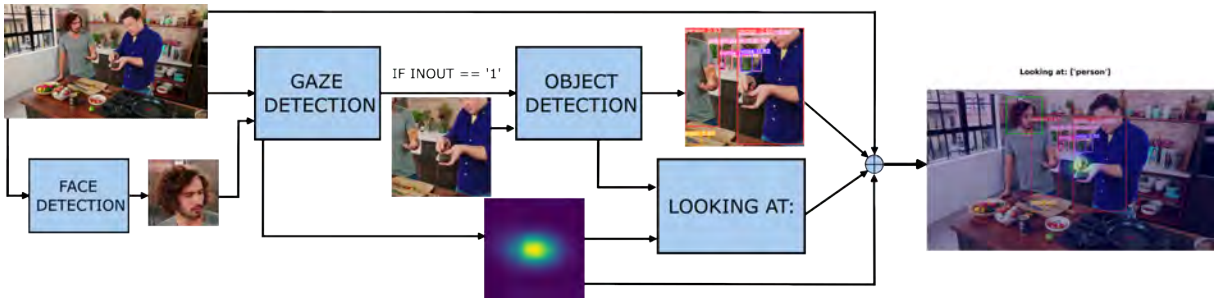


Figure 1.1: Proposed system workflow for the TFM

- Setup study: first, the camera placement will be studied. There are different options available and it is needed to establish it from the beginning.
- A face detection layer is needed to estimate the gaze orientation. Also, the parameters influence will be studied.
- As our objective is to estimate the gaze orientation and the object looked at, two sections will be added: gaze orientation and object detection algorithms.
- Then, the gaze estimation and object detection are performed and those outputs are used to estimate where the person is looking at.

1.3 Memory organization

The book that describes the TFM is distributed as follows in chapters:

- In the current chapter, an introduction is written to describe the context and scope of the work.
- Chapter 2 is added to summarize the state of art, where the different options developed within the scientific literature to achieve the proposed objectives will be discussed.
- Chapter 3 will be used to explain the system implementation: person and object detection, and gaze estimation.
- In chapter 4 the different databases used for the project will be described. Also, the metrics that measure the performance of the system developed will be defined.
- Chapter 5 includes the results given by the different systems developed, using the metrics and databases mentioned on the previous section.

- The memory finishes with chapter 6, where the conclusions and future works that could be carried out after the present project.
- Finally, a breakdown for the budget of this project in chapter 7, and the set of needed specifications for the technology transfer in chapter 8 will be added, as well as the used bibliography.

Chapter 2

State of the art

2.1 Introduction

On this section it will be studied some of the current techniques and technologies able to estimate gaze orientation and object detection, which are essential to achieve the proposed objectives.

First, the proposed setups in the state of art will be listed. Then, the gaze orientation algorithms that fit into the chosen setup will be studied. Finally, an object detection section will be added, which is split into two branches: the first one will study a general object detector and the second will be focused on face detection, which is a pre-processing layer needed on this project.

2.2 Introduction to Machine Learning

On this section the basic DL and ML nomenclature will be explained, as it is used on the book.

- Bounding Box (BB): they are imaginary boxes that contain a detected object. They have shape variability, but on this project squared BB is always used. On figure 2.2 are added some BB examples: the red lines contain the person's contour. These BB are the outputs used on this TFM.
- Inference: the process of using a trained neural network model to make predictions against previously unseen data.
- Convolutional Neural Network (CNN): it is a type of artificial neural network which is widely used for image/object recognition and classification. The convolutional layers extract the main features from the image and are fed to a fully connected layer.
- Non-maximum Suppression (NMS): is an algorithm that deletes overlapped BB after the inference.

2.3 State of gaze orientation

As mentioned in the introduction (section 1), the thesis goal is to determine the percentage of time a person is focused on a task. To achieve that objective first the setup scenario must be established: where the cameras are placed and how the videos will be recorded. Then, it is needed an algorithm that can estimate the object looked at. This objective is split into two branches:

- Face detection: a face detector was needed because it is used as an input by the gaze orientation layer.
- Gaze orientation: this algorithm has to evaluate the head pose feature and must estimate where the person is looking at with that information.
- Object detection: it must be added this section as the previous algorithm did not detect the object looked at. Its goal is to detect the objects in the frame and link its output with the gaze orientation.

2.3.1 Setups

There are two main setups to evaluate a person's gaze [1] in a unrestricted scenario using RGB video as input, which are explained below.

- **Gaze tracking in wearable devices**: in this setup (figure 2.1), the person must wear a device that has installed at least one camera on it, which will record the Point Of View (POV) [17]. The main disadvantage of this approach is that it could interference in the task performance as it is quite intrusive and could bother the patient. Besides, it will be necessary to buy a specific device, increasing the budget needed for the project and there are other concerns added, like camera calibration or battery life.

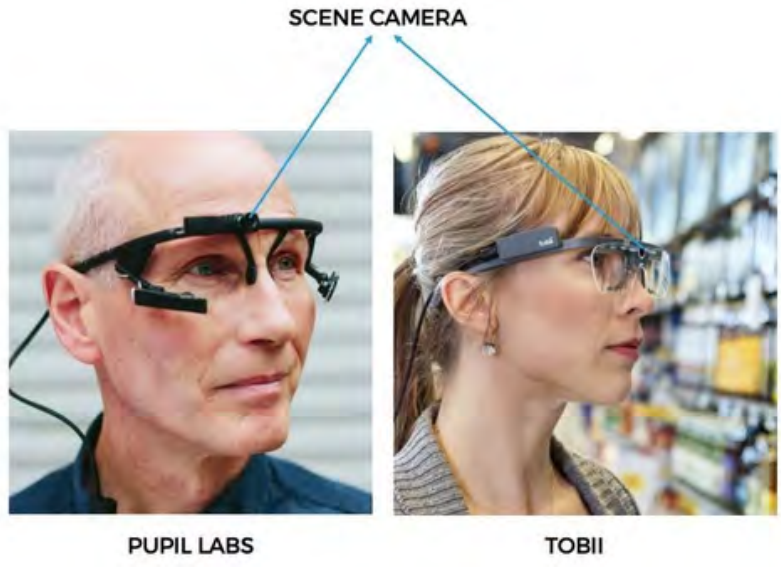


Figure 2.1: Gaze tracking with wearable [1]

- **Screen based (or remote) gaze tracking**: on this approach (figure 2.2) the camera is not worn by the user, it is located on a fixed position and will record the activities from a third-person POV. On this setup the head pose feature can be used to achieve better performance than the previous setup [7][16].

  As inconvenience, the algorithm must estimate if the person is looking at a point inside the camera's plane and, only if that premise is true, it must estimate the line of sight and which object the person is looking at among all the possible. Therefore, this setup has a huge processing workload compared to the previous solution. Also, this method often requires a pre-processing step of face/-head detection, increasing the complexity. It also can be quite ambiguous to estimate where the person is looking at if the only data available is a third person POV.

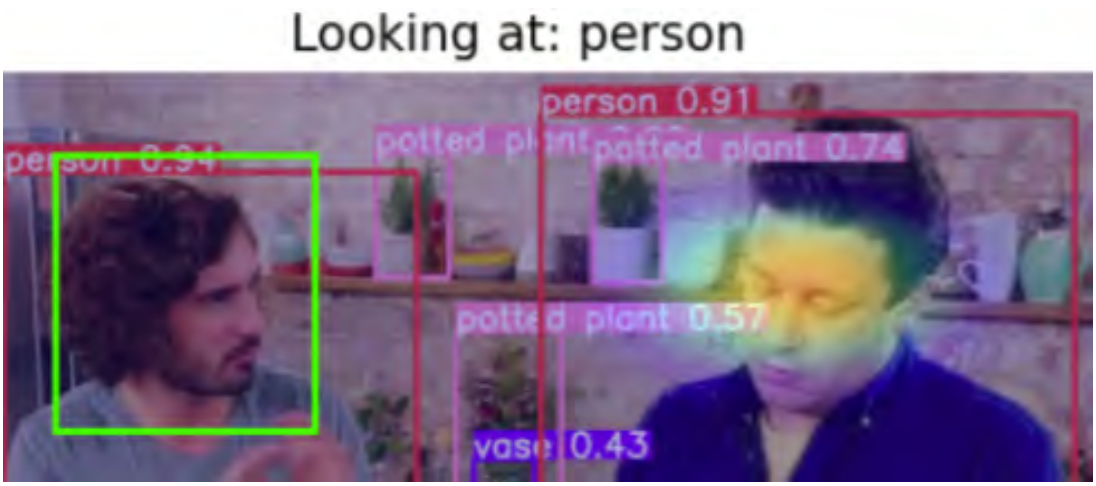


Figure 2.2: Remote gaze tracking [1]

A solution could be redundancy: if the activities are recorded from different POV (different fixed position), the ambiguity can be minimised. Another problem found using this setup was the ambiguity with depth perception: it can be quite difficult to estimate the depth in some pictures using only RGB video. In some works [7] the developers have used some techniques like using a binary image to encode the depth, decreasing the ambiguity and increasing the architecture performance and accuracy. Still, it remains a good option as it is cheaper than the previous setup and there are some algorithms that have good results [7][18][16]. Also, we could use the hardware already owned by the GEINTRA research team.

2.3.2 Algorithms

As mentioned in the setup section 2.3.1, the chosen algorithm has to be able to estimate gaze orientation in an unconstrained situation [18][16][7]. On the state of art there are different algorithms that can estimate the gaze direction on the wild based on Support Vector Machine (SVM), random forest or DL methods, which is the most recent approach [19]. If we focus on the deep learning approach, there are different architectures proposed:

- Some algorithms need a pre-processing layer to perform face or eye detection [7]. Then, if the person is looking away, there is occlusion or low resolution the face may not be detected, and without the face BB the architecture can not estimate gaze orientation. If a face detector is not used, a head detector is needed, which could lower the system accuracy as its more difficult to estimate the head pose if the person is looking away.

- In some works [16][7][18] the architecture is split into two branches. The purpose of this approach is to imitate gaze evaluation as a human does [7]: first, the observer looks at the person's head and studies on her or his head pose. Then, the observer/algorithm estimates the person's field of view and among all the possible objects, the salient object is selected [16].

- Some architectures only detect the gaze orientation but not the object looked at. In [2], it is estimated the head position in-the-wild. On figure 2.3 can be seen the output generated: yaw, pitch and roll, which are represented by blue lines. The facial landmarks are also plotted, even when there is occlusion due to head position (red point 2.3 means the marker is occluded from the camera's POV).

Figure 2.3: Gaze orientation by head position [2]

In [19] it is recorded a 360º video, with the purpose of generating a robust architecture to lighting conditions and number of subjects, but they do not estimate the object the people is looking at. Instead, they ask the people to look at a target board rather than others objects, reducing the labeling workload to only one object. The architecture is less complex than the proposed algorithm for this project, as it does not use the saliency map concept.

Those systems have a good performance on a unconstrained environment, but they are not suited for the project, as it is needed to know if the person is looking at the correct object. To achieve that, the algorithm should have a saliency map, whose objective is to highlight the most "interesting" objects in the picture (this concept will be deeply explained on chapter 3).

- Other researchers approached this problem using RGB-D data, but in [20] it is demonstrated that RGB-D degrades more than RGB when increasing the distance, concluding that depth information is not very reliable after 3.5 meters. Since the project setup has on average higher distances, it is not recommended to use depth on the project.

- Also, there is ambiguity to estimate where the person is looking at using only a frame. On [7] it is used a ConvLSTM [21], which is a type of recurrent neural network for spatiotemporal prediction, whose output depends of the actual inputs and the past states. It improves the performance compared to [22], although the most influencing parameter is the head feature.

2.4 State of object detection

This section is divided into two parts. First, a face detection chapter will be added (section 2.4.1). Then, a second section will be included (section 2.4.2), where a more general object detector will be studied, focusing on detecting which object the person is looking at.

2.4.1 Face detection

As it was explained in the setup section 2.3.1, a pre-processing layer is needed for this TFM: the gaze algorithm needs a face BB as input. On this section two algorithms have been compared:

- **MediaPipe**: is a framework for building pipelines to perform inference over arbitrary sensory data. It also facilitates the deployment on a wide variety of different hardware platforms [3] because each pipeline runs the same behaviour on different devices, allowing the developer to work on a

workstation and then deploy the app on another hardware. The main reference of MediaPipe face detector is BlazeFace [23], a lightweight and well-performing face detector tailored for mobile GPU inference. On figure 2.4 can be seen that MediaPipe's output are six facial's points: eyes, ears, face and nose position.

Figure 2.4: MediaPipe output detection [3]

This output does not perfectly fit with our approach (a head BB is preferred than his or her facial marks), and generating a BB from these values has shown more error than using a face detector. Also, it has on average lower accuracy than the following alternative.

- **Multi-task cascaded CNN**: on [4] is proposed a deep cascaded multi-task framework that exploits the inherent correlation between them to boost their performance: due to it's lightweight architecture it can be run on real-time, achieving 99 fps on GPU [4].

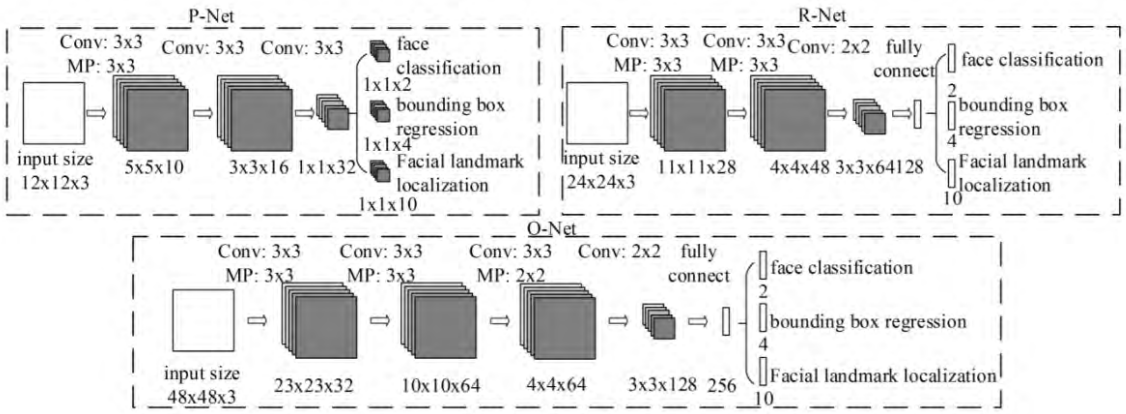


Figure 2.5: Face detector architecture [4]

The architecture has a pre-processing layer not added in the figure 2.5 that resizes the input image to different scales, generating an image pyramid in order to detect faces of all different sizes. That pyramid is used as the final input for the P-Net (Proposal-Network), then, its output is used as input in the Refinement Network (R-Net). Finally, the Output Network (O-Net) returns five facial landmarks positions (which will not be used) and a final bounding box.

The code was initially developed on TensorFlow [24], but the version used on this project was implemented by [25] on PyTorch [26] and it is trained with WIDER FACE dataset [11].

2.4.2 Object detection

On this section two different methods to perform object detection are explained. In the state of art there are two options that stand out: MobileNet [6], which is developed by Google, and YOLOv5 [5], developed by Ultralytics. In the following subsection these two algorithms will be explained.

- **YOLOv5**: first, it must be mentioned that YOLOv5 [8] does not have a published paper and the developers only showed a graphic comparing their version's performance and not the previous ones. Therefore, in this section YOLOv1's paper [5] will be used as reference as YOLOv5 inherits its philosophy. In figure 2.6 it can be seen that processing an image on this algorithm has three stages: first, the image must be resized to the input size, which in YOLOv5 is 640x640 pixels. The resized image runs the convolutional network which extracts features that are given to a fully connected layer, whose purpose is to predict the detected object probabilities and bounding-boxes.

  The first YOLO version used Darknet framework for training and inference [27]. YOLOv5 uses PyTorch framework [26] and there are several models implemented, which will be explained in the system implementation section 3.

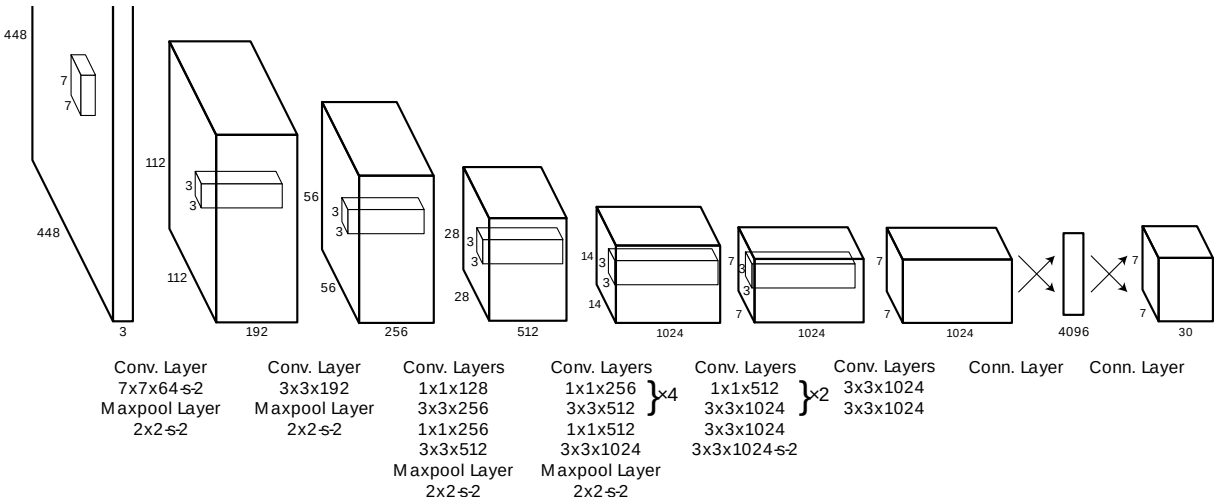


Figure 2.6: YOLOv1 architecture [5]

- **MobileNetV2**: this algorithm is focused on achieving high performance on mobile devices, reducing the memory footprint needed during inference by reducing the use of large intermediate tensors [6]. Its high performance is gotten by two features, that are also show on figure 2.7:

  - **Conv replace**: on this approach, the full convolutional operation is swapped with a factorized version, which splits it into two layers. The first one performs lightweight filtering by applying a single convolutional filter per input channel, the second one is a 1x1 convolution which builds new features through linear combinations of the input channels [6].
  - **Bottlenecks**: on this paper [6] it is used inverted residual blocks, which are more memory efficient than the original and its performance is slightly better.

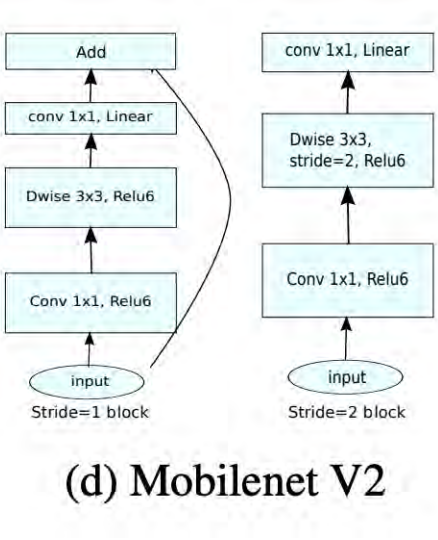


Figure 2.7: MobileNetV2 bottleneck and convolutional replace block [6]

To compare the algorithms it has been used their respective paper's results [6][8], and used the COCO dataset [10]. The comparative can be seen on table 2.1, where only the CPU performance is added because MobileNet is only tested with a mobile phone with no GPU. The mAp (Mean Average Precision) metric is explained on metrics (section 4.3).

CPU	MobileNetV2 + SSD	YOLOv2	YOLOv5n (smallest)
Delay (ms)	200	-	45
mAp	22.1	21.6	28.0

Table 2.1: YOLOvX and MobileNetV2 comparative.

Although YOLOv5 was not run on a mobile phone and there is a huge performance difference between a workstation and a flagship mobile, it must be emphasized the gap between the average precision.

2.5 Conclusions

On this TFM we have decided to use remote gaze tracking and RGB input after the study. As previously mentioned in setup section 2.3.1, it is not necessary to buy specific devices, which would increase the project budget, and it fits on the EYEFUL research project. Also, the chosen databases (section 4) explore the different options on our setup: camera height and resolution.

Focusing on the gaze orientation (section 2.3.2), the chosen algorithm uses a face detection layer and the split architecture into two branches [7][9]. Besides, we use a ConvLSTM to improve the system's performance. We do not use depth information since RGB frames are more reliable when increasing the distance.

In this project a cascaded CNN architecture [4] has been used as face detector (section 2.4.1). It was needed to adjust the output's BB, as the original was too narrow for the gaze algorithm's needs.

Finally, it has been used as the general object detector, YOLOv5 algorithm (section 2.4.2), as it has good accuracy and high performance compared to the other option. Also, it is implemented on PyTorch like the other core blocks, so it fits better than MobileNet on this project. It has been used COCO

database [10] as training data for this project, as many items used on the databases are found of interest in the TFM application.

Chapter 3

System Implementation

3.1 Proposal

In this chapter the algorithms chosen from the development of this TFM will be deeply explained. Also, the system's workflow is explained, which can be seen in figure 3.1. First, a face detection layer is needed, as mentioned in the state of art (section 2). Then, the gaze estimation is performed and if it estimates the person is looking inside the image, the object detection layer is executed in the cropped image generated around the heatmap's maximum value (this feature is explained in object detector section 2.4.2). Then, it is checked if the person is looking at an object recognized (Looking at section 3.5) and displayed the final frame.

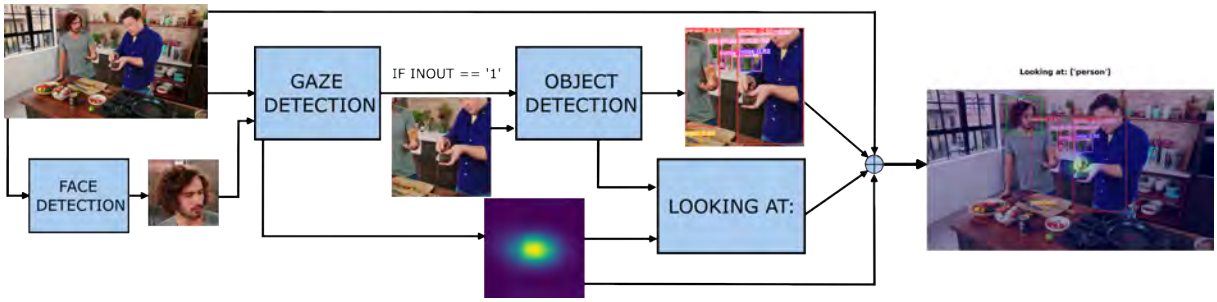


Figure 3.1: Proposed system workflow for the TFM

3.2 Gaze orientation

On this project it has been chosen a gaze orientation algorithm [7][9], which is formed with various ResNet-50 [28] and fully connected layers. The paper's goal is "to identify where each person is looking at in each frame of a video and correctly handle the case where the gaze target is out-of-frame", so it fits perfectly on the project. It was trained with VideoAttentionTarget dataset [9], that is formed with fully annotated attention targets in video.

The system inputs are a RGB image and a face BB, obtained from a face detector [4]. The algorithm returns as output the attention heatmap, indicating which zone the person is looking at and a variable called "inout", that modulates if the person is looking outside or inside the camera's plane.

The heatmap itself does not identify the object looked at, it only estimates where the person is looking. Brighter colors (orange or yellow) represent the zone estimated by the algorithm to be looked

at, and blue colors the zone not supposed to be looked at.

In figure 3.2 can be seen an example of heatmap, where the person is looking at other person's hands (as in figure 3.4 a)). Also, it does not always have the same shape: on this figure it is almost circular but it usually has a blob shape.

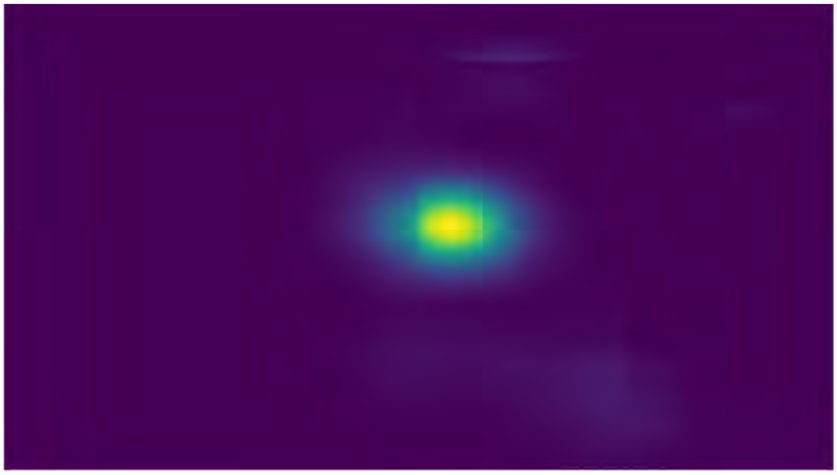


Figure 3.2: Heatmap example

The heatmap can be represented on two different ways (figure 3.3): it can be plotted on top of the image (figure b), or it can be calculated the heatmap's maximum value and then an arrow is drawn to that pixel from the face BB center (figure a). On this project the heatmap representation is the most used, since its size can fix a not accurate prediction [18].

(a) Arrow mode

(b) Heatmap mode

Figure 3.3: Heatmap output [7]

Focusing on the variable "inout", if the algorithm estimates the person is looking at an object inside the camera's plane it returns a low value (for example, figure 3.4 a), obtaining a value of sixteen). If the gaze target is out-of-frame, its output has a higher value, as can be seen on 3.4 b), with a value of one hundred ninety. After some evaluations [7], it can be assumed that the variable almost behaves as a binary output, as it does not give values between the previously mentioned. Therefore, a threshold method was implemented to decide if the observer is looking outside or not.

The system's architecture is split into two branches (figure 3.5), and each one is designed to solve its problem on its own [18][7]: the head branch must estimate the gaze pathway relying on head pose since eye tracking is not always possible due to occlusion or low resolution, and the scene branch must find the objects likely to be viewed by the person without any information about the person's POV (this

(a) Person looking inside camera's POV (b) Gaze target out of frame

Figure 3.4: Out-of-frame handle

output is also known as saliency map [16]). Then, the CONV-LSTM [21] estimates the final output (the heatmap) and at the same time is calculated if the person is looking at an object inside the frame's plane ("inout" output) as explained in "In frame?" section.

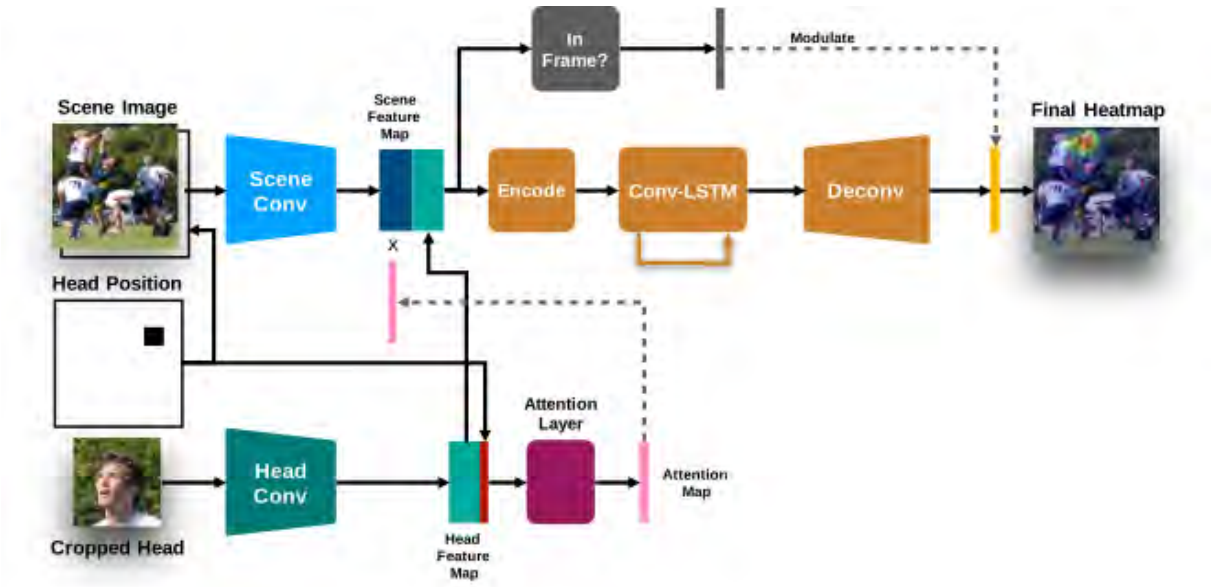
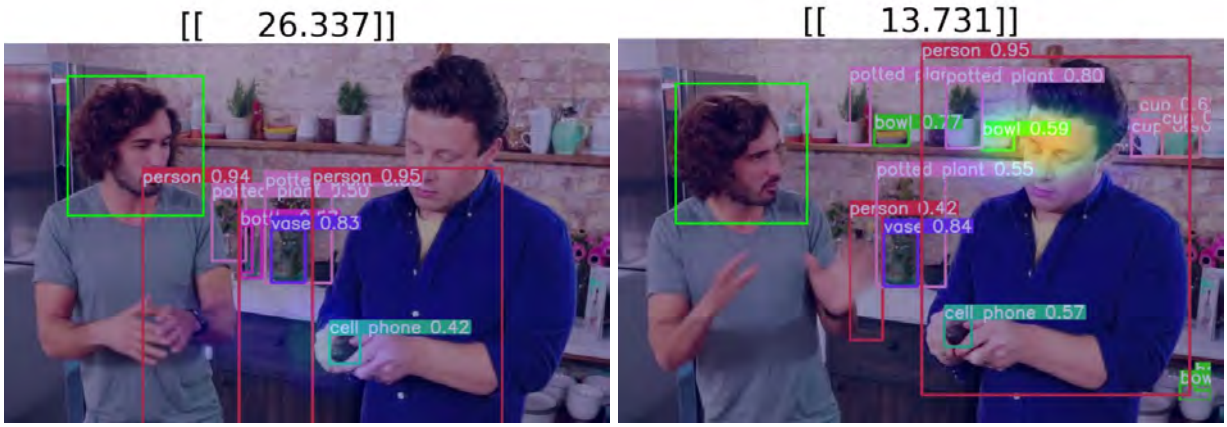


Figure 3.5: Spatiotemporal architecture for gaze prediction [7]

The saliency map can also be interpreted as a biased behaviour from the neural network since an object becomes interesting when it appears many times on the training dataset: if the system is trained with people looking at other people's faces, monitors, books... Its output will likely estimate the person is looking at one of those objects rather than an object with no "interest" or "relevance" [16]. This biased behaviour is needed since from the scene we do not know the head orientation and we have to guess which objects are the salient. If there is a salient object on the estimated gaze orientation, the algorithm predicts the person is looking at that zone. If there is no salient object in the estimated person's POV, the algorithm still gives an accurate heatmap, but not with high confidence, as explained below.

In the images 3.6 it can be seen that the output generated has a higher value on figure b) than in a). This is related to the saliency map previously explained: the algorithm is biased towards the objects used in the training sequence and this behaviour is proved in the heatmap value. When the person is looking at an object from the training sequence (figure b, looking at a person), the output has a higher

value and size. If the person is looking at an object which does not belong to the training dataset (figure a, the person on the right is holding an avocado), the heatmap has on average a lower value and size.



(a) Low heatmap value (b) High heatmap value

Figure 3.6: Algorithm bias towards trained objects rather than uninteresting objects

This is due to the scene, that has a priority to estimate gaze to salient object as TVs, people faces and other objects from the training dataset. As our project is not focused in those objects, it can worse the neural network performance since in the databases used there are prominent objects that are considered not interesting or salient by the neural network. A solution to this problem could be fine tuning the CNN, but at the moment there is not enough data to do it. Besides, it is needed to annotate the gaze GT, increasing the workload. Therefore, this task is proposed as a future work.

This architecture also has problems with depth perception, as it can be ambiguous to estimate where the person is looking at in only one frame. For example, in figure 3.7 there is some ambiguity. Where is he looking to? The algorithm estimates he is looking at the person, but he could be looking outside the camera's plane. To summarize, it is difficult to estimate where the person is looking at in one frame, even for a human.

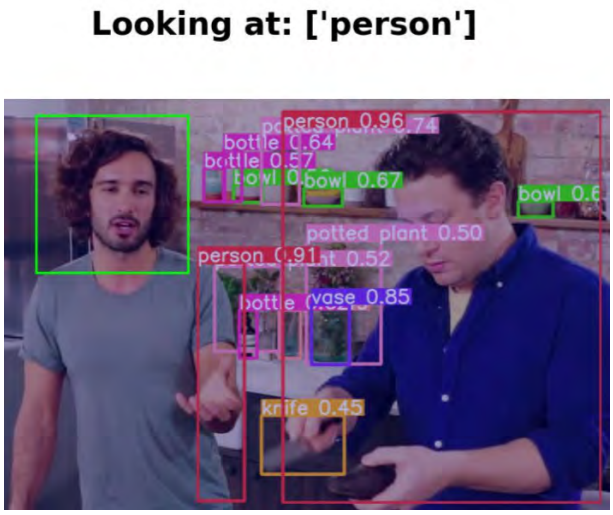


Figure 3.7: Where is he looking at?

Another concern is the influence of the BB in the system's precision. On [7][16], it is used a GT face location, but as our approach uses an automatic head detection algorithm, there were some doubts about its impact on the system's performance. On [16], researchers affirm that using an automatic head

detection system instead of using a GT head location only worse the Area Under the Curve (AUC) 1%, probing that our approach is viable.

3.3 Face detector

As already mentioned in the gaze orientation section 2.3.2 and the conclusion section 2.5, a face detector (pre-processing layer) is needed, which will add more complexity to the system. The chosen algorithm is a cascaded CNN architecture [25][4], that is trained with [11].

The main concerns about using a face tracker in the application of interest in the TFM are the following:

- If the person is looking away from the camera or there is a partial occlusion, the face may not be detected by this layer and the system can not estimate gaze orientation [18]. This problem could be solved using a head detector instead of a face detector. Then, the impact on the system's accuracy should be studied, as it would influence the head features gotten in the head branch, figure 3.5. Another solution could be holding the last heatmap value. This approach can have a good performance, but it depends of the video used.

- On figure 3.8 a) can be seen the original BB size of the face detector. As already mentioned on the gaze estimation section 3.2, one branch gets features from the cropped head image, and if it is not given enough information about the head's surroundings, the accuracy has proven to decrease significantly.

  Therefore, the BB was increased for this project as it can be seen in figure 3.8 b). Increasing the BB also added more concerns, which are explained on the following paragraph.

(a) Default BB size (b) Increased BB size

Figure 3.8: Cropped image by head detection layer

- Sometimes, the pre-processing layer may crop the object the person is looking at, and in that situation the object can worse the head features gotten. Although the gaze estimation algorithm will return an output, it can affect the system accuracy.

Finally, another problem found is handling a wrong face detection. In that case, the system will fail to estimate the gaze orientation and the object looked at, decreasing the whole system's accuracy. To avoid wrong face detections, the following restriction was added: the algorithm only uses the highest confidence face BB. This feature also simplifies the algorithm as we only have to handle one input per frame and increases the system accuracy because we avoid false face detections. As final conclusion, the

face detector was added to the system as an auxiliary layer and it has become the most critical output, as a wrong BB can not be fixed.

3.4 Object detector

As previously mentioned on the state of art section 2.4, the object detector algorithm used for this project is YOLOv5 [8][5]. It has been trained with COCO dataset [10], as many of its items appear on the videos used on the TFM. There are some objects of interest in the TFM which do not belong to this dataset and could be useful for our project as kitchen cabinets or napkins. These objects are quite difficult to detect on a RGB image, as some of them have shape variability. On this project it has been used a static label to indicate the cabinet's position, as the camera is not moving on the recording due to its fixated position.

On YOLOv5 there are different models, varying the number of convolutional layers, the FC layer size and the inference time (figure 3.9). Even so, the algorithm can still perform real-time detection due to its lightweight architecture. This feature is the main reason this architecture is known for: high speed with relatively good accuracy.

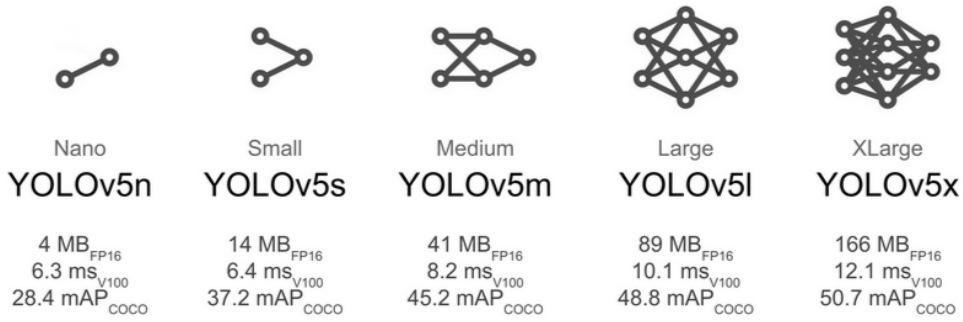
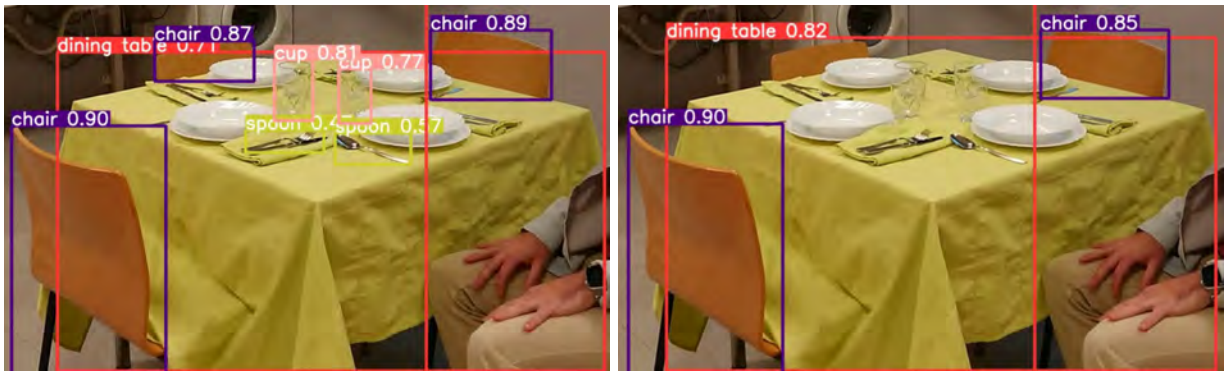


Figure 3.9: YOLOv5 models [8], COCO dataset

In image 3.10 (figure a) can be seen the output generated by the largest model, XLarge. It struggles or fails to detect small objects like silverware, and if they are detected, they usually have low confidence. However, it can detect with high confidence bigger objects as the tables, chairs, cups, people...
In the small model (figure b), there is almost no detection of small objects and the biggest are detected with high confidence, but lower than the XLarge model.



(a) XLarge model (b) Small mode

Figure 3.10: YOLOv5 example

On the computer used for this project it takes around 40 milliseconds to perform the inference on the XLarge model, allowing a 25 fps rate and real time performance, as previously mentioned on the SOTA section 2.4.2. As it was able to detect the smallest objects and has a real-time performance, the XLarge model was used on this project.

After choosing the model used on this project another problem was found: the input size of YOLO is relatively low compared to the image resolution in the databases, forcing the object detector to resize the image and losing performance due to object size and shape deformation. To improve object detection performance, the following idea was proposed: generate a cropped image from the raw frame, centered in the maximum value of the heatmap. Its shape is YOLO input's size (640x640 pixels) and it is used as input for the object detection layer. This approach improves the system performance, as the layer can detect small object that were not detected before. In figure 3.11 the improvement can be seen, as there are more small objects detected than in 3.10.

(a) YOLO output (b) Final frame

Figure 3.11: YOLO output when the cropped image is used

If the heatmap's maximum value is close to the frame's raw edges the new image would need padding to fill the YOLO's input size. On this project it is used a different approach: the center of the cropped image is moved towards the image center, as can be seen in figure 3.12.

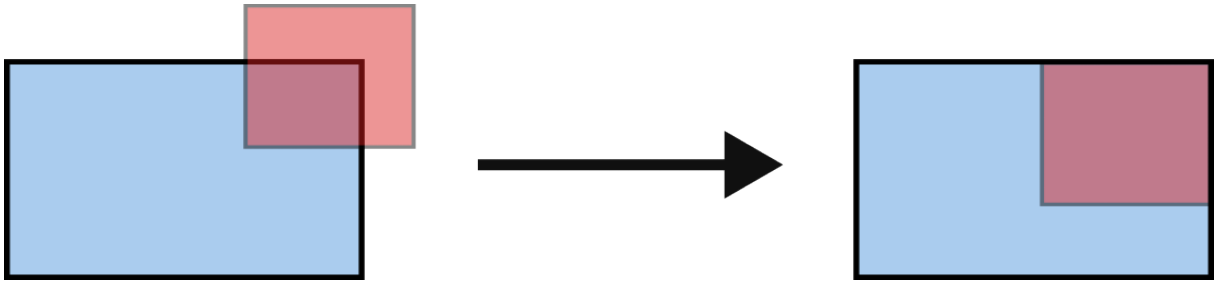


Figure 3.12: Cropped image algorithm

3.5 "Looking at" algorithm

On figure 3.1 it was mentioned a layer called "Looking at". The purpose of this section is to estimate if the person is looking at a detected object. Two methods have been developed to perform this task:

- The first method relies on evaluating the heatmap's maximum value and calculate if the pixel is inside an object's BB. It is quite simple to develop but there are some concerns: the main purpose of using the heatmap output instead of its maximum value is to fix a not accurate prediction [18], so this solution could worse the output although the heatmap generated is correct.

- The other solution proposed was implementing an IoU, which also has other problems. First, a BB must be generated around the heatmap's maximum values. To perform this, it is needed to normalize the heatmap's values and use a threshold to generate the blob. Then, the square BB is generated around the blob's contour (yellow line). In figure 3.13 an example can be seen, where two different threshold values have been used on the same heatmap.
  If the used threshold is low, the generated BB does not correctly represent the heatmap (figure a). If the threshold value is high, another problem is found: the BB generated is quite small compared to the BB generated by the object detector (figure b), and the method almost behaves like the heatmap's maximum value method.

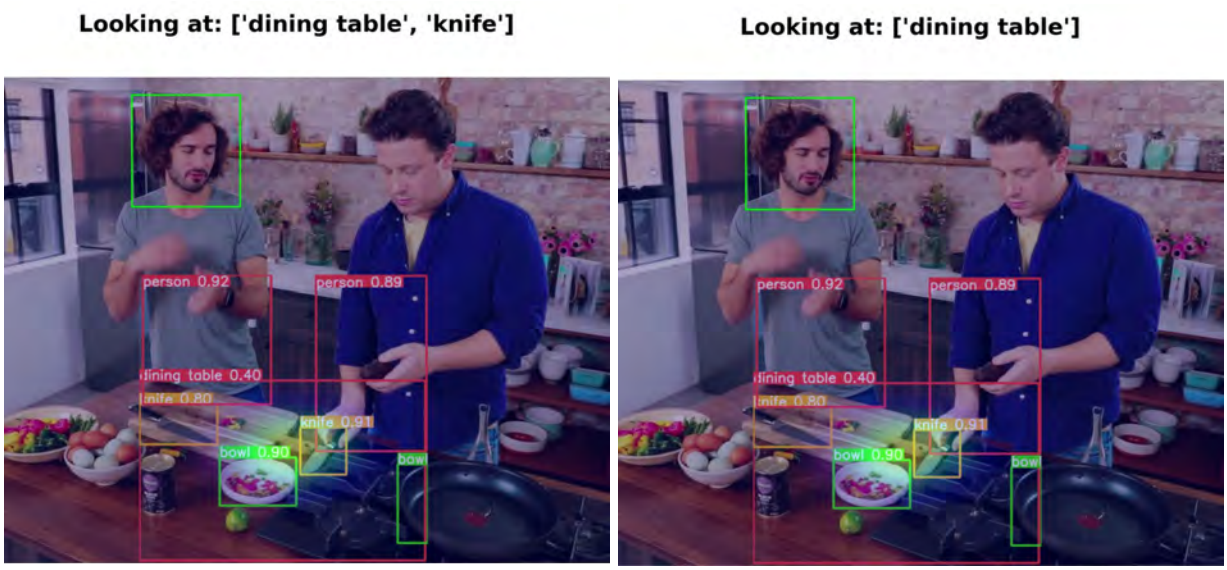
(a) IoU threshold: 40 percent, not fit enough (b) IoU threshold: 70 percent, small bounding-box

Figure 3.13: Heatmap normalization concerns

In figure 3.14 some examples are shown. In figure a) and b), the observer is looking at the knife that is close to the person's hand. In this example, the IoU method detects the knife and the dining table, while the pixel method only detects the table.

In this project the Intersection over Union (IoU) method has been used with a 70 percent threshold. The previous example proves the IoU method is more robust than the pixel method, as the heatmap can fix low accuracy outputs [18], although there are not many differences between both methods' performance.

Finally, three different labels have been implemented to indicate the object being looked at. If the person is looking outside the plane, the following text will pop on top figure "Looking outside the plane". If the person is looking inside the plane and there is object recognition the text displayed will be "Looking at: (object list)", but if there is no object recognition, the following text will pop up "No object recognized".



(a) IoU, threshold 70 percent (b) Heatmap's maximum value

Figure 3.14: YOLO output when the cropped image is used

3.6 Conclusions

On this section the system workflow (figure 3.1) has been explained.

- First, the gaze orientation algorithm has been studied [7][9], showing their inputs (face BB and the RGB frame), their outputs ("inout" and the heatmap) and the concerns about it.
- Then, the face detector section is added [4][25]. Different concerns are stated, as a fail on this layer means a fail in the gaze estimation (if the face detection fails, the gaze outputs will be random). To avoid false face detection, the following condition is added: the layer only returns the highest confidence BB, simplifying the whole system workflow. This feature also fits on the research project, as there should be only one person performing the ADL.
- The object detector used on this TFM was YOLOv5 [8][5]. First, the model selection was discussed, being the chosen option the largest model, XLarge model. Then, the cropped image feature is explained: a cropped image is generated around the heatmap's maximum value, with the purpose of improving the object detector performance. The cropped image shape is YOLO input's size, so there is no need to reshape the original frame and it could improve the confidence on the small objects.
- The last section was "Looking at" (section 3.5). On this section the different algorithms used for detecting if the person is looking at a detected object are explained. The chosen algorithm was implementing a IoU, whose size can fix a not accurate prediction [18].
- As final conclusion, it is ambiguous to estimate where the person is looking at, even for a human. Temporal information (memory from previous frames) can help in some frames to evaluate where the person is looking at, but there will still be frames where it is ambiguous to estimate where the person is looking at

Chapter 4

Evaluation methodology and setup

On this section the datasets used on this project will be explained. They are split into two sections: databases used to train the system or to test its performance. As already mentioned on the introduction (section 1), this project is focused on people performing ADL, so the databases are related to that subject.

Also, the different metrics used for quantitative results will be explained and, as we do not have GT on every dataset used on this TFM, a qualitative result section will be added.

4.1 Datasets used on training

The training datasets are VideoAttentionTarget [7], COCO [10] and WIDER FACE [11]. The first is used to train gaze estimation, returning the "inout" and the heatmap output previously explained on section 3.2. The second is used to perform object detection, already explained on section 3.4: once the system returns the heatmap output, it is needed to detect the object looked at (section 3.5), if there is any object on that zone. Finally, the third is used to detect faces, as it is needed as the first layer of gaze detection (section 3.3). Quantitative results of these databases are added on the results section 5.

4.1.1 VideoAttentionTarget

This dataset [9] was generated using videos from different sources: live interviews, sitcoms, movie clips..., gathering 1331 tracks. In these tracks the GT of the head bounding boxes were labeled, as well as their estimated gaze target and the option to indicate if the person was looking outside the camera's frame. This generated over 110,000 in-frame gaze targets and around 55,000 out-of-frame annotations.

Over 30,000 frames were holed out to test its performance, ensuring no overlap of frames between train and test split, allowing the investigator to measure the model capacity for new scenarios. The following images, in figure 4.1, belong to the training split.

4.1.2 COCO dataset

COCO dataset [10] has a total of 2.5 million labeled instances in 328k images. The objects' labels were limited to entry-level categories commonly used by humans: for example, a German shepherd and a golden retriever would be labeled as "dog". Similar images were deleted to grant diversity on the dataset.

(a) Star Wars film (b) How I Met Your Mother (TV show)

Figure 4.1: VideoAttentionTarget dataset [9]

Partial occlusion was considered interesting by the developers, as it would be beneficial for training AI algorithms for many real-world applications. The dataset has been split in the following sets:

- Train: 19G, 118k images
- Val: 1G, 5k images
- Test: 7G, 41k images

On this project all the classes from this dataset are not used, but it includes many items useful for the system (see figure 4.2): fork, knife, spoon, cup, dining table, bowl and chair.

(a) Bowls, spoons, table... (b) Fork, bowl

Figure 4.2: COCO dataset [10], validation set

4.1.3 WIDER FACE

WIDER FACE [11] is a face detection benchmark which has around 32,000 images with 393,000 faces labeled. It is a subset of the WIDER dataset [29], whose images were collected from Google and Bing search engines. Then, similar images or those without a human face were deleted, generating WIDER FACE [11]. The GT BB containing the face is required to tightly contain the forehead, chin, and cheek (see figure 4.3). Therefore, it was needed to increase the output size BB for our system as explained on face detector section 3.3.

(a) Press conference I

(b) Press conference II

Figure 4.3: WIDER FACE [11], validation set

4.2 Datasets used on testing

There are different POV and resolutions with the purpose of exploring and studying the different options available on the chosen setup: record the patient from a fixed position on a third person POV. Two of them (TUM kitchen [30] and ETRI [31] dataset) are from other research teams and the last one (EYEFUL [32]) was recorded on our own and it is not published yet. As there is no GT, these databases can not be used for training or quantitative results.

4.2.1 TUM Kitchen

The recorded sequences [30] consist of several instances of a table-setting task in a natural kitchen environment, provided as a 384x288 pixel RGB image sequence. All subjects perform the same task with some variations, as explained below.

Some subjects perform the activity in a robot-way (transporting the objects one by one) and others in a natural way, grabbing as many objects as they can at once. The camera distribution can be seen on figure 4.5, represented by red blobs. The height of the camera can be guessed from figure 4.4, in which it is higher than in the other datasets used (on this dataset the cameras are close to the ceiling).

Figure 4.4: TUM kitchen image

The object distribution goes as follows: the placemat and the napkin were placed in location A, the cup and plate in an overhead cupboard at location B and the silverware was in a drawer between A and B. Finally, the objects were placed on location C or D.

The videos have the following nomenclature: $ID_{ij}Subject_kTask_m$, which is explained below:

- i: '0' is the objects are dropped at location C, else '1' and they are dropped at location D.
- j: it is the number's sequence. There are 13 videos with i=0 and 8 videos with i=1.
- k: person's ID. Four people were recorded on this dataset.
- m: indicates the chore (stt=set the table) and how the activity is performed (robot/human way).

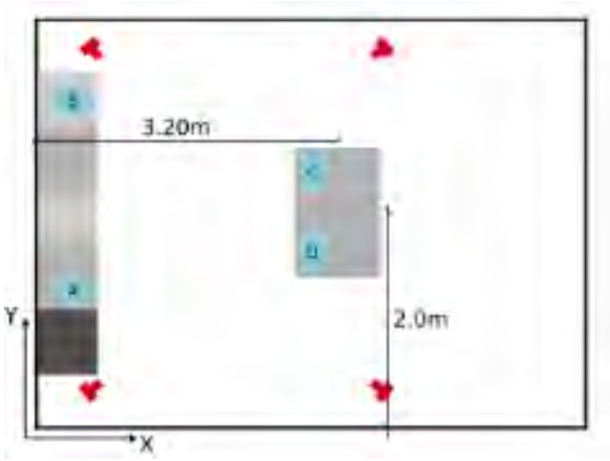


Figure 4.5: TUM kitchen distribution

4.2.2 ETRI

This database [31] is focused on recognition of ADL from a service robot's POV. There are 55 activities recorded: eating food with a fork, pouring water into a cup.... and they are recorded in the same apartment, whose floor's distribution can be seen on figure 4.6, left side.

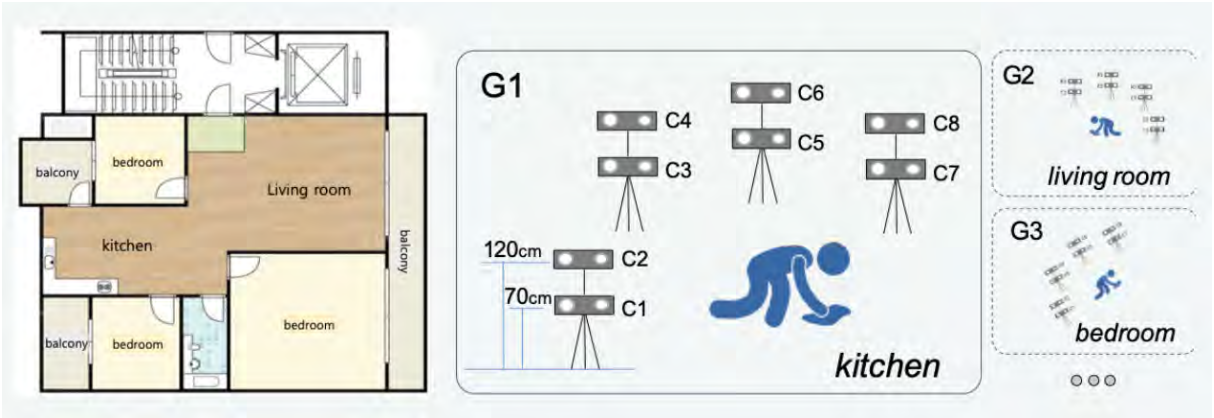


Figure 4.6: ETRI's distribution

The subject's age is divided into two groups: half of the recorded people are elderly (who are assumed to be the main users of these service robots) and the other half are people on their twenties.

It is recorded with 2 Kinect sensors (RGB resolution 1920x1080), whose distance between the subjects and the camera varies from 1.5 meters to 3.5 meters. As the robot's height is supposed to be relatively low, the cameras were placed at two altitudes (70 and 120 cm) in the same spot, as it can be seen on figure 4.6, right side. The low height will be a problem in our approach, since there are some activities where it can not be seen properly the object looked at by the person.

The following activities have been used from this database: eating food with a fork, pouring water into a cup, cleaning up the dining table, looking around for something and using a remote control. Finally, it is needed to blur the faces in the images for personal information protection.

The videos have the nomenclature $A_iP_jG_kC_m$, whose explanation in tackled within the list that follows below:

- A_i: activity's number.
- P_j: person's ID.
- G_k: room's identification.
- C_m: camera number.

4.2.3 EYEFUL-DB

These videos were recorded for EYEFUL project [15]. The objective of these recordings is to generate data to develop an automated system with the purpose of evaluating clinical assessment and determine the percentage of functional limitation of a person.

P1	P2	P3
P4	P5	P6
P7	P8	P9

Figure 4.7: URJC kitchen distribution. Cameras are on P6, P7 and P9 position

The videos were recorded in Universidad Rey Juan Carlos (URJC), where it is emulated a standard apartment. The cameras were placed on different positions: on figure 4.7 the floor distribution is shown and in figure 4.8 two frames recorded from P7 and P9 are added. Also, in some recordings the camera placed on P9 was moved to P6. Three different activities were recorded, but only two of them were used in this project: setting the table and sweeping the floor, which were performed by different people in the kitchen.

On setting the table task (figure a 4.8), the person has to grab the items from the cupboard (P2 position on figure 4.7) and place them on the table (P5 position). On the sweeping floor task (figure b 4.8), the person has to clean the kitchen area with a broom. The height of the camera is over 1.8 meters: higher than ETRI [31] but lower than TUM kitchen [30].

4.3 Metrics

On this section different metrics used on this project are explained. As mentioned above, quantitative and qualitative results are used due to lack of GT in some databases. Before writing down the equations used, it is needed to define the basic nomenclature:

- TP - True positive
- FP - False positive
- FN - False negative

(a) Set the table task, EYEFUL-DB (b) Swept the floor task, EYEFUL-DB

Figure 4.8: EYEFUL-DB example

- TN - True negative
- P - Total positive results
- N - Total cases

4.3.1 Quantitative metrics

On this section will be explained the different metrics to obtain quantitative results, that will be then used in section 5.

- **Precision**: percentage of your predictions are correct.

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

- **Recall**: measures the model's ability to detect positive samples.

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

- **AP (Average precision)**: its the area under the precision-recall curve. It computes the average precision value for recall value over 0 to 1. This metric is usually used when the dataset does not have their classes equilibrated and properly classifying the positives is important [33].
- **mAP (Mean average precision)**: mean average precision over all categories. In the results it may appear the following nomenclature [34]: AP@[.5:.95]. It corresponds to the average AP for IoU from 0.5 to 0.95 with a step size of 0.05. Also it might have the following nomenclature: AP@.75, that means the AP with IoU=0.75.

- **Distance**: distance between the annotated target location and the prediction given by the pixel of maximum value in the heatmap with image width and height normalized to 1 [7].
- **True Positive Rate**:
$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{4.3}$$
- **False Positive Rate**:
$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{4.4}$$
- **AUC (Area Under the Curve)**: is calculated as the area underneath a curve that measures the trade off between true positive rate (TPR) and false positive rate (FPR) at different decision thresholds. It is considered a good algorithm if the AUC is higher than 0.8, and an AUC of 0.5 means the output is random. It is a good choice when all the classes have roughly equal number of observations.

4.4 Conclusions

On this section the different databases used on this TFM and the different metrics used on the result section has been explained.

Regarding the databases used for training, graphics and tables will be added to prove their performance on their test splits, measured in the metrics explained above.

The other databases used on this project do not have the people's gaze labeled, so the results can not be evaluated with a quantitative method. In the result section some frames will be added and the general opinion related to the configuration used.

Chapter 5

Results

On this section it will be shown the different output generated by the whole architecture. As mentioned on evaluation chapter 4, there is no GT on all the databases used, so the results section has to be split into quantitative and qualitative results.

5.1 Quantitative results

On this subsection the test results from the used algorithms will be added. Also, the timing results will be added.

5.1.1 Experiments with VideoAttentionTarget

The test results are shown on table 5.1.2. It had to be split into four splits, as the PC could not load the whole data without crashing. In the fifth column is added the mean average of the four sets, in the sixth the results published by the developers, and in the last column the human results are added [7].

Split	**AUC**	$L^2 distance$	**"inout" AP**
First	0.8654	0.1380	0.9032
Second	0.8481	0.1415	0.8292
Third	0.8806	0.0909	0.7157
Fourth	0.7813	0.2000	0.7783
Average (Ours)	0.8439	0.1435	0.7783
Average [7]	0.860	0.134	0.853
Human [7]	0.921	0.051	0.925

Table 5.1: VideoAttentionTarget dataset, quantitative results

Although there is a gap between the human AUC and our results, it is a good result for an algorithm. In the "inout" row there is a bigger difference between the human results and ours, and it is a huge concern on this TFM: a false negative of "inout" variable worse the whole system's performance because if the algorithm estimates the person is looking outside the plane, there is no object detection.

Finally, the difference between splits is interesting, as there is a huge gap between splits results. The split was not performed according to the folder difficulty, it was based on their folder size with the purpose of equilibrating the splits.

5.1.2 Experiments with YOLOv5/ COCO dataset

There is no GT of the dataset used on this TFM, so the only quantitative results we can show are those mentioned by YOLOv5 developers [8]. On the following table 5.2 there are some benchmarks from the different models ("X" is the biggest model and "N" the lowest).

Model	Size (pixels)	mA I 0.5:0.95	mAP I 0.5	Speed V100 b1 (ms)	Param (M)
YOLOv5x	640	50.7	68.9	12.1	86.7
YOLOv5m	640	45.4	64.1	8.2	21.2
YOLOv5n	640	28.0	45.7	6.3	1.9

Table 5.2: COCO dataset, quantitative results

As mentioned on 3.4, there is a huge gap between the fastest and the biggest model. As our project is not related to real-time performance, using the largest model is the best option.

5.1.3 Experiments with WIDER FACE

On figure 5.1 it can be seen the Precision-Recall graphic (AP measure) on a medium and hard set, outperforming the results from other state-of-the-art methods. On the first image (medium set) the algorithm increases the AP measure by 20 percent versus the second algorithm, and in the second image (hard set), it doubles the second best result, achieving an AP of 60 percent versus the second best result which is 31,5 percent.

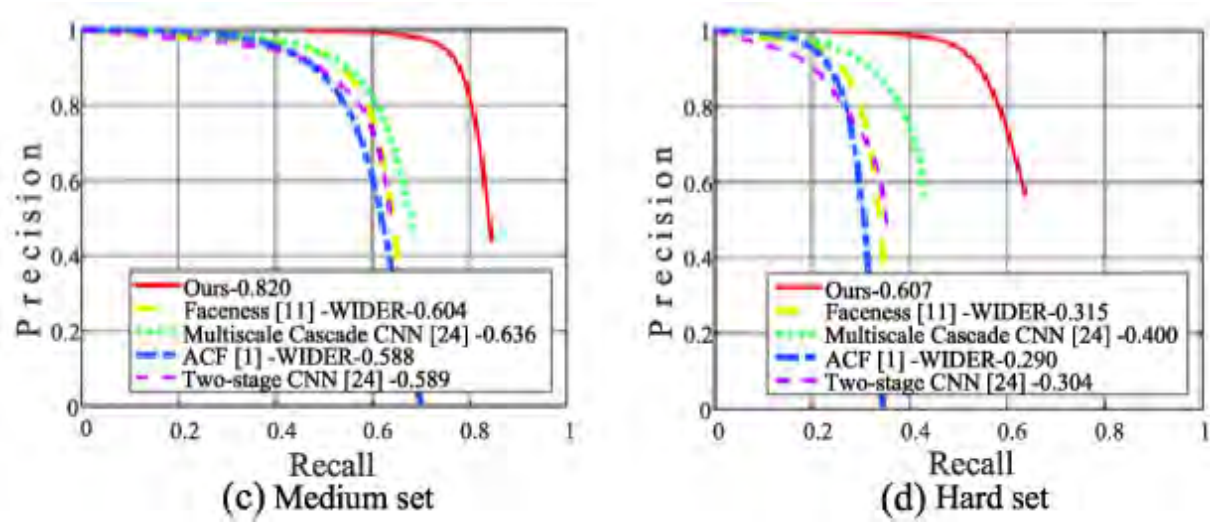


Figure 5.1: Evaluation on two subsets of WIDER FACE [11]

The results showed on the previous figure state that this architecture and algorithm has good performance compared to other options.

5.1.4 Timing

Finally, these are the average timing results (table 5.3) of the whole project, that was executed on this PC (hardware resources section 7.1). All the timing results are on milliseconds.

Resolution	Gaze infer.	YOLO infer.	General YOLO infer.	Save image	Plot image
1920x1080	63	125.1	132.6	170	35.7
348x288	15	62,8	200	150	33.2

Table 5.3: Timing results from ETRI and VideoAttentionTarget databases

- **Gaze algorithm inference**: average gaze algorithm inference time. It always has similar values, around 65 milliseconds.
- **Inference (YOLO prediction)**: time used by the YOLO model. The inference time on a test split (125 milliseconds) takes longer than only executing the YOLOv5 algorithm.
- **General YOLO inference**: YOLO inference time plus cropping the image and the "Looking at" algorithm.
- **Plotting/save image**: saving the image takes more time than just plotting the image because accessing the disk is more time-consuming.
- **Face detection**: it is not added on the figure because the algorithm is executed before the gaze algorithm, but performing inference a frame takes around 20 milliseconds.

The algorithm takes 2fps if the frames are saved on disk and almost 3fps if the image is plotted, so the algorithm is far away from real-time performance. Most of the videos were recorded at 30fps, so executing the algorithm on a 1 minute video takes approximately 15 minutes.

Regarding the TUM kitchen database, the inference time is lower but the object detection takes much longer than it should, lasting more than 0,130 seconds to perform only the NMS in some frames. As YOLOv5 input's size is bigger than the resolution, it is needed to resize the raw frame, deforming the object's proportions and generating too many BB. This behavior decreases the timing of the workflow, and it takes over 2 fps if the frames are saved on disk (it is the most used method).

5.2 Qualitative results

On this section there will be added images that show the algorithm performance in the following databases: TUM Kitchen [30], ETRI [31] and EYEFUL database, that is not published yet so there is no paper to cite.

5.2.1 Experiments with EYEFUL-DB dataset

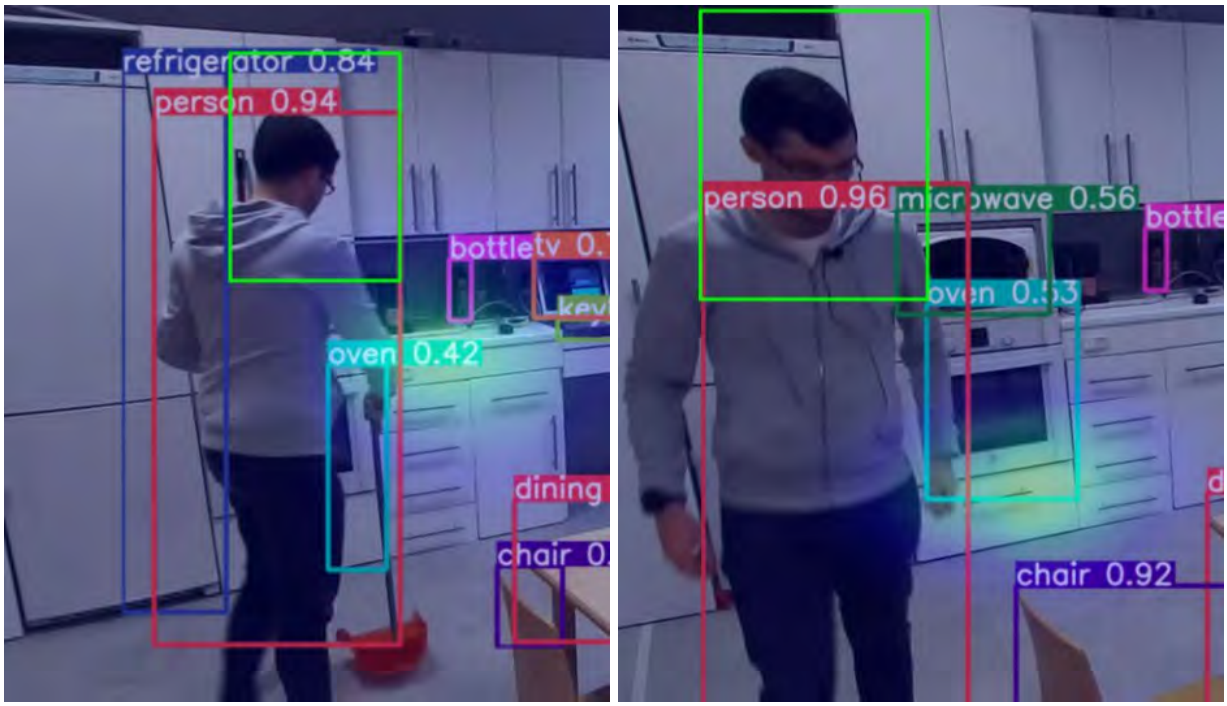
These videos belong to the research project mentioned on the introduction (section 1). Although there are many videos recorded, only frames from four videos will be added, as the algorithm has the same behaviour on all of them.

The gaze algorithm performs better when the person is handling objects close to the table (figure 5.2 a) and b) P6 position).

(a) Correct face detection and gaze estimation (b) Correct face detection and gaze estimation

Figure 5.2: EYEFUL-DB database performance

If the person is close to the cupboard section (P1-3), the algorithm often has problems with depth estimation, although it does not happen on every frame. On figure 5.3 a) the algorithm estimates the gaze correctly, but on figure 5.3 b) the depth estimation is wrong, giving as output a heatmap that indicates the person is looking at an object positioned on his backward.



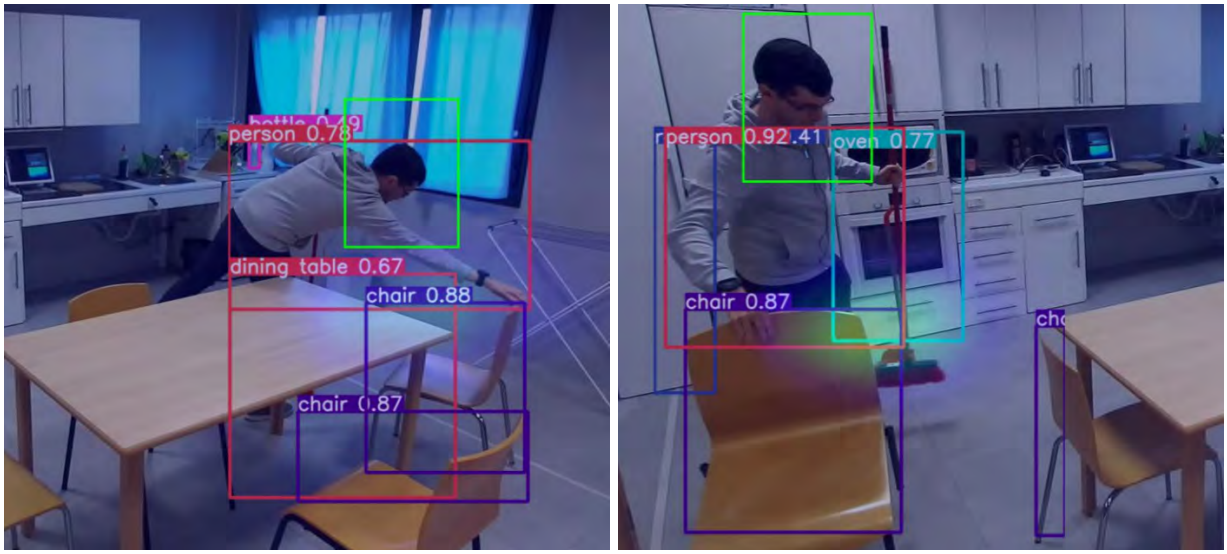
(a) Correct depth estimation (b) Wrong depth estimation

Figure 5.3: Algorithm performance on P4 position

As conclusion for these images, when the person is facing the camera on P4 position, the algorithm is biased towards failing depth estimation. But, if the person in that position is looking at the table or facing backwards, the algorithm often estimates the gaze correctly (figure 5.2 b)), so not all the estimations from that position are wrong. My hypothesis is that most errors occur when the person is transitioning between tasks.

There are also some problems with ambiguity. For example, on figure 5.4, it is difficult to estimate where the person is looking at. On figure a), the person is looking away from the camera, so we only have the head pose, and in this situation I do not think there is enough information to estimate the gaze with confidence, although the heatmap output is quite accurate. On figure b), the person is transitioning between tasks, and on those situations the person does not always look at the object handled in that moment, they look at the item related to the next task (on this case, the next task is grabbing the other chair).

These figures (figure 5.4 a) and b)) also prove that the algorithm uses the heatmap value as a confidence metric: on figure a) the heatmap is barely visible, while on figure b) the yellow color is clearly visible. This is due to the head features gotten: in figure a) the head branch can not get many information about it, so the heatmap output is not as confidence as in figure b), where the head features can be easily extracted.



(a) Due to body and object occlusion the output has low confidence

(b) The transition frames between actions usually are amibiguous

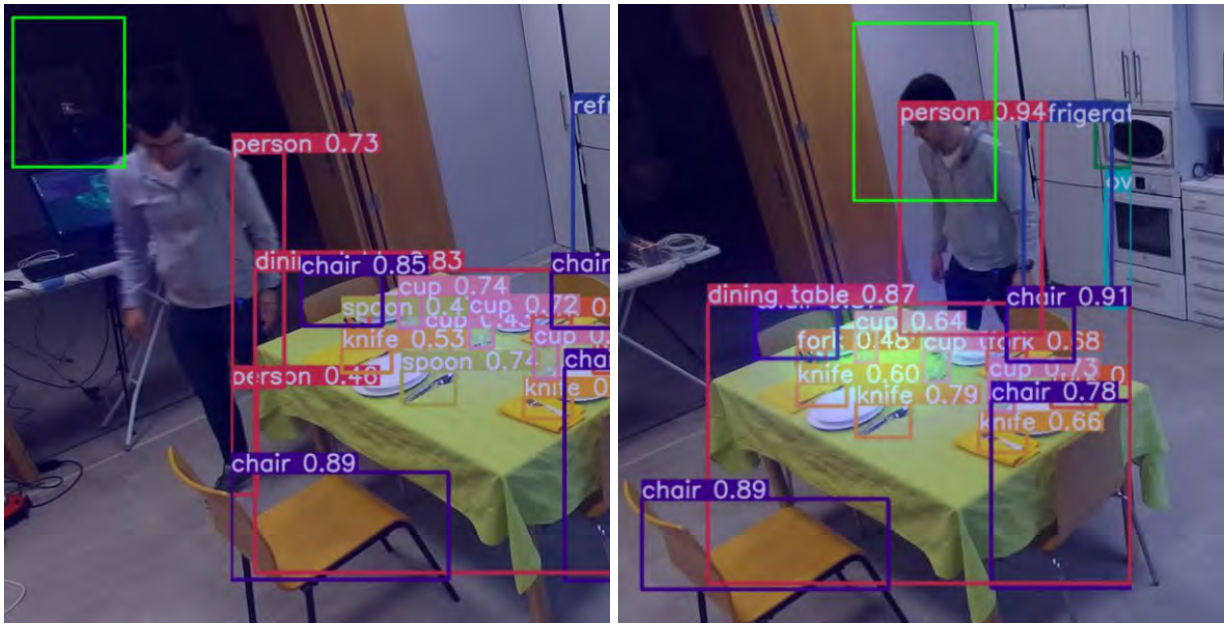
Figure 5.4: Ambiguous situations for the gaze algorithm

Regarding the face detector, its performance worsens when the person is not in a front facing position towards the camera. As it can be seen on figure 5.5, two similar frames get a quite different output.

This behaviour was only found in zones with low luminosity, as can be seen on figure 5.5 a). On the result section is only added one frame with a wrong face BB, but in all the videos recorded from P9 and P6 position the same wrong detection was repeated on that zone.

Another problem found on the face detector layer was that, if the person is looking sideways from the camera, the algorithm usually miss the face BB (this behaviour is shown in TUM kitchen result section 5.2.3).

On figure 5.5 a) and b) can also be seen the objects detected by YOLOv5 when the task is finished. It can detect many small items like glass cups, knifes, spoons...so, compared to the original YOLOv5 output added on the YOLOv5 (section 3.4), the cropped image feature has improved the system's accuracy related to object detection.



(a) Wrong face detection

(b) Objects detected on final frame, setting the table task

Figure 5.5: Setting the table, recorded from P9 position

Another concern found was that the hair occlusion worsens the face algorithm performance. On figure 5.6 it can be seen that this layer is more likely to detect a masked person placed far away from the camera (figure a)) than a long hair person (figure b)).

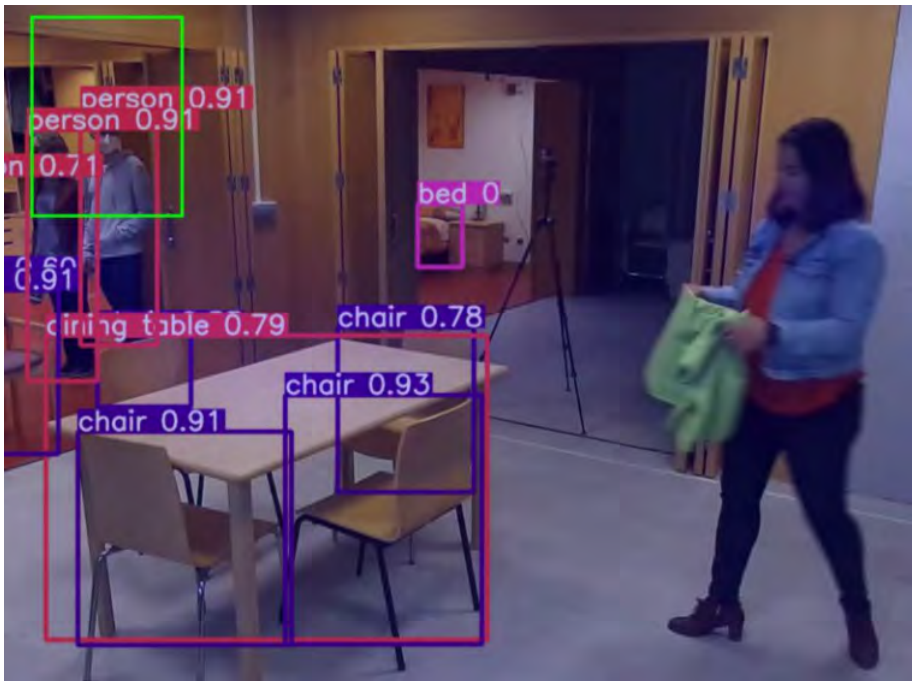


Figure 5.6: [Hair occlusion avoids closer face detection

After the evaluation, the following conclusions are stated within a qualitative analysis specifically oriented to the application of interest in this TFM:

- The distance between the camera and the person is appropriate, as the small objects are detected and it can record the whole room.

- There is a problem related with depth perception. When the person is moving between tasks on position P4, the depth estimation fails more frequently than in other positions.

- It may be interesting to use different camera POV for redundancy, but it may increase the system complexity without achieving better results: it would be needed an algorithm that could fix a disagreement between the different camera outputs.

- As we have had many problems with zones outside the kitchen, I would recommend closing the doors in the next recording section. Also, it is recommended to have good lighting condition on the room recorded.

- I would recommend to pull the person's hair back with the purpose of avoiding hair occlusion of the face.

- It can detect masked people, but it is more difficult for the face detector. Regarding the gaze estimation algorithm, it still had good performance because the head features can still be gathered.

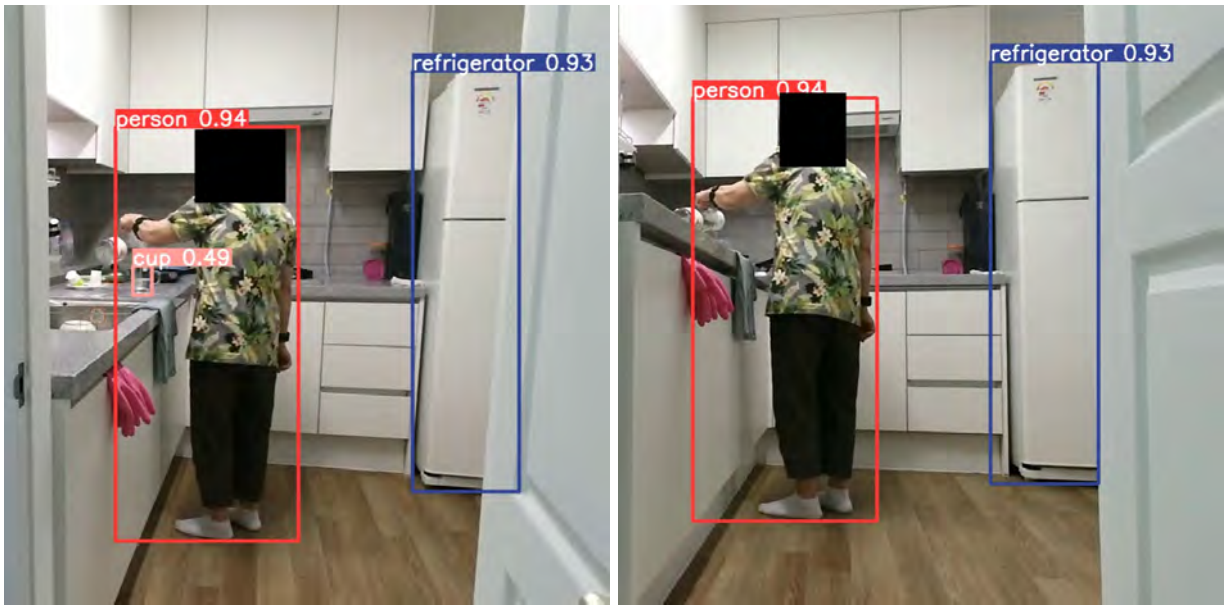
- A head detector instead of a face detector should be tested. Many frames are not executed because there is no face detection because the person is looking away from the camera. Also, in many frames where the person in not looking at the camera, the heatmap is accurate enough, so it may not decrease the system's performance. Even so, it is quite difficult to generate a GT in those situations because it is ambiguous to estimate where the person is looking on that frame.

5.2.2 Experiments with ETRI dataset

The database owners require to blur people's faces. As it is needed to see the face to estimate the gaze, it is useless to add any image related to gaze estimation, so I will comment about the camera POV concerns: camera height and object occlusion.

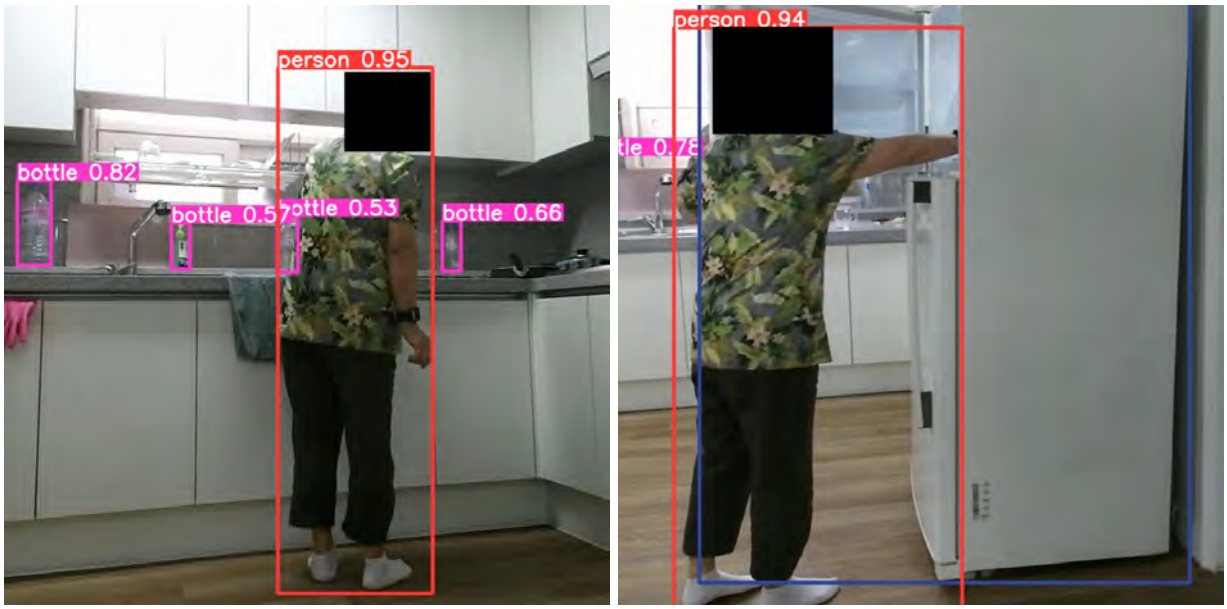
- Camera height: in many videos from this dataset the low height of the camera robot was a concern (the camera height is 70 and 120 cm). In the right image (figure 5.7), the glass can not be detected due to the low camera height, while in the left image it is detected.

- Body and object occlusion: this problem is found in all the databases used for this project. In some POVs and frames the person's body or an object breaks the camera's line of sight and the object looked at by the person can not be detected.



(a) Camera height: 120 cm (b) Camera height: 70 cm

Figure 5.7: Camera height influence, ETRI database



(a) Partial occlusion by the person (b) Complete occlusion by object

Figure 5.8: POV influence in ETRI database

Finally, the distance between the camera and the patient is appropriate for this TFM (it varies from 1.5 meters to 3.5 meters), as it allows object recognition in small objects and has enough distance to record the whole room.

In many video categories, the POV of the low camera could not detect the object handled by the person, so a higher altitude is desirable. Also, there should be more than one camera recording the action: this would minimize the occlusion problem.

5.2.3 Experiments with TUM Kitchen dataset

The main difference between this database and the other is the resolution (384x288 pixels versus 1920x1080) and the camera height: it is closer to the ceiling than in the other databases. Even so, the system has a similar behaviour to the EYEFUL database, as it is explained below.

On figure 5.9 can be seen some examples of good algorithm performance. On figure a), the person is looking at the item placed on the table with high confidence and accuracy, although the object detector can not detect many items.

On figure b) can be seen some examples where the algorithm estimates the depth correctly when the person is close to the cupboard, unlike the EYEFUL database 5.2.1.

Regarding the object detector, it struggles to perform the inference. The low resolution forced the layer to resize the frame raw, deforming the frame and decreasing the layer performance. For example, on EYEFUL database the silverware was detected and on this database it was not in any video. Also, the cropped image algorithm can not be executed, as the YOLOv5 input size is bigger than the database resolution.

(a) Gaze estimation when the person is looking at the table

(b) Depth estimation

Figure 5.9: Gaze estimation, TUM kitchen database

Regarding the face detector, it has the same problems that in EYEFUL database 5.2.1: if the person is in front of the camera, there is an accuracy detection. If the person is looking away, moving aside or there is partial occlusion, the face BB is still accurate, but the detection is almost random. For example, on figure 5.5, two contiguous frames have the opposite behaviour: the first frame has no detection and the next one has a BB.

This database has the same problems when estimating where the person is looking while transitioning between tasks. On figure 5.11 a) and b) are shown some examples: the person is walking towards the cupboard to grab another item and the algorithm does not estimate the depth correctly. On this figure

(a) Frame 286, no face detection (b) Frame 287, has face detection

Figure 5.10: Camera height influence, ETRI database

there is also an ambiguity problem: it is difficult to estimate where the person is looking at because it is likely the person is looking at something outside the camera's line of sight.

(a) Ambiguous gaze estimation (b) Ambiguous gaze estimation

Figure 5.11: Wrong gaze estimation transitioning between task

As final conclusion for this database, it is possible to state different questions hereby listed and commented below:

- The camera height and POV are correct, as it allows to record the whole action.
- The main concern about this database is the low resolution. It is not a problem for the gaze algorithm or the face detection, but it is for the object detector: the database frame resolution is lower than the YOLOv5's input size and it has to resize the original frame, deforming the item shape and decreasing the layer performance.
- It does not have the same problems with depth perception when the person is close to the cupboard, but it still fails the gaze when the person is transitioning between task, as in EYEFUL database results 5.2.1.

- The face detection layer has the same behavior than in EYEFUL and ETRI database: if the person is not front facing the camera, the algorithm miss many detection, being almost a random behaviour: two similar frames (or contiguous frames) have opposite BB detection.

5.3 Comparison of the results

On this section all the conclusions mentioned above will be measured and abstracted in order to extract the most important analysis from the work here developed.

- The face detector had the same behavior on all the databases: it struggled to detect the faces when the person was not looking directly at the camera. It also had problems with low luminosity zones and people with long hair, although it could detect masked people. I also found a weird behaviour on EYEFUL database: the BB were bigger than in other datasets, maybe due to the camera's height. As it is needed a face detection to execute the other algorithms, in many frames it could no be estimated the gaze, decreasing the system's accuracy.
- Regarding the gaze estimator, it had many problems with depth estimation on EYEFUL database. This behaviour was only found on those videos, although in all databases the same problem was found: there is an error cluster when the person is transitioning between tasks, as there is ambiguity about where the person is looking at. Once the person is focused on the task, the algorithm usually has accurate outputs.
- The general object detector layer had a good performance on ETRI [31] and EYEFUL, as it could detect small objects as silverware. On the other hand, it struggled to detect objects on TUM kitchen [30], as the frame's resolution was lower than the input's CNN size and it had to resize the raw frame, increasing the inference time and decreasing the system's performance, as small items as silverware were not detected.
- Regarding the "Looking at section", I think it had a good performance on all the databases as I have not found frames where the labels were wrong.

Chapter 6

Conclusions and Future Works

This chapter, just at the end of the memory, is dedicated to clarify the final conclusions extracted from the whole TFM project, and moreover, to state the proposed future works that can improve it.

6.1 Conclusions

The algorithm developed for this TFM has been tested on different databases and the following conclusions can be stated:

- Regarding the gaze algorithm (section 3.2), in my opinion it has a good accuracy estimating where the person is looking at (it has a good AUC metric), but it should improve the "inout" estimation as the mAP is low compared to the human result. After some testing, the "inout" errors are usually good for this TFM, because it usually fails to estimate when the person is looking outside the camera plane due to lack of depth perception. The object detection layer is only carried out if the algorithm estimates the person is looking inside the camera plane.

  If the algorithm estimates the patient is looking outside, there is no object detection, and it can not be estimated which item the patient is looking at, which is a worst case that estimating the person is looking inside when he or she is looking outside in my opinion.

- After working on this TFM with recordings from a third POV, I think it is the best option (setup section 2.3.1): it is cheaper than using a wearable device and it does not bother the patient.

- The camera POV and angle are important. If the height is low the camera will not be able to record which object the patient is handling or looking at. After the study, I would recommend lifting the camera above the patient's head. Regarding the camera angle, at least two cameras are needed because the person or the object can block the line of sight between the camera and the object handled or being looked at.

- The object detection layer (section 3.4) has issues detecting small objects like forks, spoons, vases... Also, it is difficult to detect objects with many of shape variability, like drawers (it can be open, full of objects or empty...) or napkins, that also have a lot of color variability.

- The face detector layer (section 3.3) was added as an auxiliary layer for the gaze algorithm, but it is the most important layer on this TFM. If there is a wrong face detection, the system will fail the prediction, there is no chance to fix a low accurate prediction, as the gaze estimator algorithm [18].

- Regarding the IoU of the BB, the best mode is using the normalized heatmap instead of using the pixel with the highest value. This method allows to fix not accurate predictions from the object or gaze layer.
- As final conclusion, I think that all the layers have a good performance, although there is still some room from improvement. Some of the possible ideas are presented in the following section.

6.2 Future Works

It is also important to extract some ideas from the global TFM development for further works. The most important ones are listed below:

- We should study the influence of using a head detector instead of a face detector. It would allow to estimate where the person is looking even when the patient is looking away form the camera, but it may decrease the system performance because the predictions without head features can decrease the accuracy significantly [7].
- The “inout” AP should be increased, as it is a critical output on this system and it is relatively low compared to the heatmap AUC (table 5.1).
- Fine tuning the object detector and the gaze estimation algorithm to increase accuracy and detect objects related to the EYEFUL project.
- In the gaze estimation architecture, replace the ”Scene Conv” with an object detector (if possible, the fine tuned version mentioned on the previous future work). This change in the architecture should improve the system’s performance on our specific project, as the ”Scene Conv” used at the moment is focused on a more general gaze estimation.
- Another future work related to object detection is finding an object detector that does not struggle to detect small objects as YOLOv5 [8] does.
- This algorithm can only uses a face BB per frame. It may be interesting to handle more than one face per frame for future projects.
- The setup used on this TFM could take advantage of the redundancy added by more than one camera recording: sometimes the camera can not detect the object looked due to body or object occlusion, but maybe other camera could do it. It may be interesting to develop an algorithm that uses the results from different POV.

Chapter 7

Budget

This chapter offers an estimation of the total cost of the present project. The following sections include the breakdown of all expenses split according to their type and origin, giving the sum of all the costs and the total budget in the last section.

7.1 Hardware Resources

The present project has been accomplished with a PC that has an Intel core i7-5820K CPU 3.3GHz, with a GeForce GTX 980 and 32 GB memory RAM. This equipment was provided by GEINTRA research team and it is valuated in 1000 €. Also two cameras have been used on this TFM to record the EYEFUL-DB 4.2.3, that it is not published yet.

Concept	**Unit. Price**	**Quantity**	**Subtotal**
PC	1.000,00 €	1	1.000,00 €
Zed2i	600 €	1	600 €
RealSense D435	500 €	1	500 €
TOTAL			2.100,00 €

Table 7.1: Hardware resources expenses.

7.2 Software Resources

The software that has been used to carry out this project has been free licensed software or the University of Alcalá has provided a student license for free to its community.

- PyCharm Community edition.
- Microsoft Office 365 programs.
- LaTeX.
- Inkscape.

7.3 Human Resources

As this project has been carried out by an engineer, the human resource cost is reduced to the hours of work needed for this engineer (see Table 7.2). This TFM was developed in 8 months, working 350 hours. Technician expenses are als added, including on this hours the setup installation and administrative processes.

Concept	Hour Price	Quantity	Subtotal
Engineer	100,00 €	350	35.000,00 €
Auxiliary technician	70,00 €	20	1.440,00 €
TOTAL			36.440,00 €

Table 7.2: Human resources expenses.

7.4 Material Execution Budget

The sum of the previous breakdown expenses, including all the resource execution expenses to accomplish this task is presented in the following table, 7.3.

Concept	Cost
Hardware Resources	2.100,00 €
Software Resources	0,00 €
Human Resources	36.440,00 €
TOTAL	38.540,00 €

Table 7.3: Budget of the global material execution budget.

7.5 Contract Expenses

In this section, the respective expenses to the use of the installations where the project has been carried out, fiscal charges, financial expenses, administrative fees and derivatives of the project control obligations are included. The industrial benefit is also included. The estimation of all these costs have been done applying a 22% surcharge on the global material execution budget of the sum in table 7.3.

Concept	Cost
22% of the global material execution budget	8.478,80€

Table 7.4: Contract expenses.

7.6 Contract Earnings

The earnings set for the present project are calculated as a percentage of the sum of the material execution budget and the contract expenses, setting this percentage to the 10% of the total sum of the budget, as the table 7.5 shows.

Concept	Cost
Total sum of the budget	47.018,80€
10% of the material execution budget	4.701,88€

Table 7.5: Contract earnings

7.7 Global Budget

This section is dedicated to provide the summary of the different budget sections previously defined in order to obtain the global cost of this project. The results are shown in the following table 7.6.

Concept	Cost
Material execution budget	38.540,00 €
Contract expenses	8.478,80 €
Contact earnings	4.701,88 €
TOTAL (no VAT)	51.721.56 €
VAT (21%)	10.861,53 €
GLOBAL BUDGET	62.583,10 €

Table 7.6: Global budget.

Chapter 8

Specifications of the project

To carry out this project, certain types of hardware and software resources have been used throughout the analysis and comparison work.

8.1 Hardware resources

As mentioned on hardware resources section 7.1, the following devices has been used on this TFM:

- PC with an Intel core i7-5820K CPU 3.3GHz, a GeForce GTX 980 and 32 GB memory RAM.
- Zed2i
- RealSense D435

8.2 Software resources

Multiple software resources and programs have been used to finish the project.

- PyCharm Community edition was used to develop the algorithm.
- Microsoft Office 365 programs (Excel, SharePoint, Teams, OneDrive, Word, PowerPoint) have been used for multiple tasks in this project.
- LaTeXhas been used to write the memory of this project.
- Inkscape. It is a Free and open source vector graphics editor for GNU/Linux, Windows and macOS.

Bibliography

[1] S. Mallick, "Gaze tracking," https://learnopencv.com/gaze-tracking/, november 2019.

[2] R. Valle, J. M. Buenaposada, and L. Baumela, "Multi-task head pose estimation in-the-wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2874–2881, aug 2021. [Online]. Available: https://doi.org/10.1109_2Ftpami.2020.3046323

[3] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, "Mediapipe: A framework for building perception pipelines," 06 2019.

[4] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, oct 2016. [Online]. Available: https://doi.org/10.1109%2Flsp.2016.2603342

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2015. [Online]. Available: https://arxiv.org/abs/1506.02640

[6] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2018. [Online]. Available: https://arxiv.org/abs/1801.04381

[7] E. Chong, Y. Wang, N. Ruiz, and J. M. Rehg, "Detecting attended visual targets in video," 2020. [Online]. Available: https://arxiv.org/abs/2003.02501

[8] "Yolov5 github," https://github.com/ultralytics/yolov5.

[9] E. Chong, Y. Wang, N. Ruiz, and J. M. Rehg, "Detecting attended visual targets in video," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[10] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2014. [Online]. Available: https://arxiv.org/abs/1405.0312

[11] S. Yang, P. Luo, C. C. Loy, and X. Tang, "Wider face: A face detection benchmark," 2015. [Online]. Available: https://arxiv.org/abs/1511.06523

[12] "Ot definition," https://www.wfot.org/about/about-occupational-therapy.

[13] "Amps definition," https://www.innovativeotsolutions.com/about/.

[14] C. G. Bravo and M. P. R. Pérez, "Rjc master class," master Class RJC.

[15] m. a. t. buena pregunta, "Multisensorial analysis of human activity for diagnosis and early detection of functional limitations (eyeful)," hhttp://www.geintra-uah.org/eyeful/es/informacion, mEMORIA CIENTÍFICO-TÉCNICA DE PROYECTOS.

[16] R. Adria, K. Aditya, V. Carl, and A. Torralba, "Where are they looking?" 2015.

[17] T. Fischer, H. Chang, and Y. Demiris, "Rt-gene: Real-time eye gaze estimation in natural environments," 09 2018.

[18] D. Lian, Z. Yu, and S. Gao, "Believe it or not, we know what you are looking at!" 2019. [Online]. Available: https://arxiv.org/abs/1907.02364

[19] P. Kellnhofer, A. Recasens, S. Stent, W. Matusik, and A. Torralba, "Gaze360: Physically unconstrained gaze estimation in the wild," 2019. [Online]. Available: https://arxiv.org/abs/1910.10088

[20] S. S. Mukherjee and N. M. Robertson, "Deep head pose: Gaze-direction estimation in multimodal video," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 2094–2107, 2015.

[21] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," 2015. [Online]. Available: https://arxiv.org/abs/1506.04214

[22] E. Chong, N. Ruiz, Y. Wang, Y. Zhang, A. Rozga, and J. Rehg, "Connecting gaze, scene, and attention: Generalized attention estimation via joint modeling of gaze and scene saliency," 2018. [Online]. Available: https://arxiv.org/abs/1807.10437

[23] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, "Blazeface: Sub-millisecond neural face detection on mobile gpus," 2019. [Online]. Available: https://arxiv.org/abs/1907.05047

[24] "Tensorflow main page," https://www.tensorflow.org/.

[25] "Face detector github," https://github.com/timesler/facenet-pytorch.

[26] "Pytorch main page," https://pytorch.org/.

[27] "Darkent main page," https://pjreddie.com/darknet/.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[29] Y. Xiong, K. Zhu, D. Lin, and X. Tang, "Recognize complex events from static images by fusing deep channels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[30] M. B. Moritz Tenorth, Jan Bandouch, "The tum kitchen data set of everyday manipulation activities for motion tracking and action recognition," 05 2010.

[31] J. Jang, D. Kim, C. Park, M. Jang, J. Lee, and J. Kim, "Etri-activity3d: A large-scale rgb-d dataset for robots to recognize daily activities of the elderly," 2020.

[32] RJC and UAH, "Eyeful dataset," 2022, aDLs.

[33] "Measuring performance: Auprc and average precision," https://glassboxmedicine.com/2019/03/02/measuring-performance-auprc/.

[34] "map (mean average precision) for object detection," https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173.

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR