Eldians

VISIONVISTA

EMPOWERING SIGHT ENABLING INDEPENDENCE

PREPARED FOR
MoroccoAI2023

PREPARED BY
ZAOUG Imad
EL BAHA Ali
LOURARI Yahya

Foreword :

This report outlines the work undertaken as part of our participation in the Morocco AI 2023 hackathon, centered around assisting individuals with visual impairments. Our primary objective was to design smart glasses aiming to, to the extent possible, replace the visual function of users by helping them detect their surroundings. This ambitious project focused on the application of various technical methods and tools to create an innovative assistive device. Throughout the following sections, we will detail the different steps of our approach, highlight the obtained results, and share the technical challenges overcome. We express sincere gratitude to our collaborators, the organizers of the Morocco AI 2023 hackathon, and our development team. We hope this report will contribute to the advancement of visual assistance technologies and inspire benevolent initiatives in the field of artificial intelligence.

Summary

Chapitre 1

Introduction

1.1 Context :

The project we are working on is part of the MoroccoAI 2023 hackathon, an event dedicated to integrating artificial intelligence into innovative solutions. Taking place online from December 11th to 17th, 2023, this hackathon aims to encourage innovation and collaboration within Morocco's emerging AI ecosystem.

The overarching goal of the hackathon is to stimulate participants' creativity to solve real-world problems in specific areas such as Education, Healthcare, Environment, Finance, and Customer Services. This initiative provides a unique opportunity to create solutions based on the latest advances in artificial intelligence, particularly in the field of generative AI.

1.2 Project Overview :

Our project within the hackathon focuses on using Language Models (LMs) to develop smart glasses to assist visually impaired individuals. The approach involves connecting the LM to a speech-to-text model and a text-to-speech model. This combination will transform visual information captured by a 3D object detection model via a camera into actionable data.

The goal for the first version of this project is to train the model using prompts, relying on data extracted from the object detection model. This will enable the system to describe the visually impaired person's environment, maintain a conversation, and assist them in their daily activities.

1.3 Expectations for the First Version :

For this initial iteration, we aim to achieve several objectives. Firstly, we seek to successfully train the model using relevant prompts to ensure an accurate

description of the environment. Additionally, we hope the model will be capable of sustaining a smooth conversation based on data extracted from the object detection model.

This initial version will serve as a foundation for future improvements, particularly concerning the efficiency of environment description and the quality of interaction between the model and the visually impaired user. We are excited to actively participate in the hackathon and explore the limitless possibilities of generative AI to create an innovative and meaningful solution.

Chapitre 2

Final Product Design : Exploring Inspired Models and Sources of Inspiration

2.1 Introduction to Model Features :

2.1.1 Global Objective of the Model :

The overarching goal of our model is to create an integrated solution based on artificial intelligence, specifically Language Models (LM), to enhance the quality of life for visually impaired individuals. Our model aims to provide comprehensive assistance by converting the visual perception of the environment into actionable data, facilitating empathetic and enriching interaction.

2.1.2 Vision for the Final Version and the First Prototype Version :

Final Vision :

The final version of the model envisions the integration of a pair of glasses equipped with a microphone to detect user speech and speakers to provide responses. To enhance the 3D object detection capability, we are also considering the installation of two lenses to assess the depth of objects. However, for simplifying the assembly and improving 3D detection precision, an alternative could involve using a stereoscopic lens arrangement, providing a better perception of depth.

In addition to these components, the glasses will have the ability to connect to the internet via Wi-Fi. This connection will facilitate communication with the Language Model (LM) installed on a separate server. Given the difficulty or impossibility of integrating powerful GPU units into the glasses, the server will

act as the model's analyzer and responder, while the glasses will serve as the intelligent user interface.

2.1.3 Model Overview and Prompting Strategy

In the figure, you will find a simplified presentation of the model and its main functionalities.

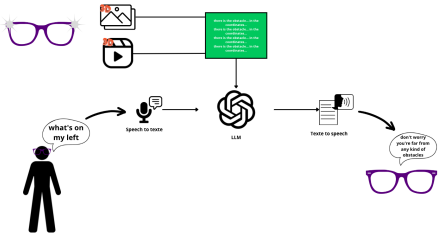

FIGURE 2.1 – VisionVista Simplified Model

It is important to note that we will rely on prompts to align the model with our expectations by providing it with information about the surroundings, explaining each detected object with its relative position to the camera, and addressing the person's requests. We will organize all this in a structure similar to applying the Chain of Thought prompting method.

For the voice-to-text and text-to-voice models, we have quite performant models that we can use. The real work will be in the Language Model (LLM) part, where we will engage in prompt engineering to organize environmental information and enhance the results.

2.2 3D Object Detection :

The 3D object detection is a pivotal element in our project. In this section, we will discuss an example of a 3D camera, inspiring 3D models, and other approaches for extracting information about the depth of objects in an image.

The initial step in this process is conducting a thorough study of the individuals involved and their daily activities. This is crucial for defining the classes of the object detection model, considering the specific needs of these individuals. For instance, defining a "road" class is essential, and training the model through transfer learning ensures it can detect this class and alert the person if they are near a road.

Another critical aspect is that the model must be real-time to perform its job effectively. A model that is not real-time would not be beneficial, as it would introduce delays in extracting information from images and additional delays for other parts of the system, rendering its use ineffective.

This condition narrows down the choice of models to consider, but fortunately, there are models that meet these requirements, such as the ZED 3D Object

Detection model from the ZED camera, which we will delve into further in the next section.

Regarding the type of camera to use, it is imperative to opt for 3D cameras, typically equipped with two lenses, like the example of the ZED 2i. This is because models that extract depth information work with RGBD images. However, if this type of camera is unavailable, an alternative is to use an object detection model coupled with a depth estimation model as a replacement for the 3D object detection model. This latter model works with RGB images, which poses no issues in terms of compatibility with most cameras that generate RGB images.

2.2.1 Example of 3D camera : ZED 2i

The ZED 2i is a stereo depth camera that has the capability to capture three-dimensional images in real-time. With its two lenses, the ZED 2i enables stereoscopic vision similar to human perception, providing accurate depth information for each captured image. Additionally, the ZED 2i generates depth maps that quantify the distance between the camera and objects in the scene. It also incorporates real-time 3D object detection models. However, it's important to note that these specific models are designed to work exclusively with the ZED 2i camera. Despite this limitation, the ZED 2i remains an excellent example of a 3D object detection camera due to its advanced features and ability to provide precise and real-time depth information.

FIGURE 2.2 – Caméra ZED 2i

2.2.2 ZED 3D Object Detection :

The ZED 3D Object Detection model is a technology developed by Stereolabs, specializing in stereoscopic vision systems and depth perception. This model aims to detect and locate three-dimensional objects in scenes using stereo cameras and depth sensors.

The model employs computer vision and deep learning techniques to analyze stereo images and depth data, detecting objects in the scene. Visual and spatial features of objects are extracted, allowing for the identification of their classes and determining their positions and orientations in space.[**?**]

The advantage of this model lies in its ability to remember detected objects from previous images, assigning them a unique identifier as long as they do

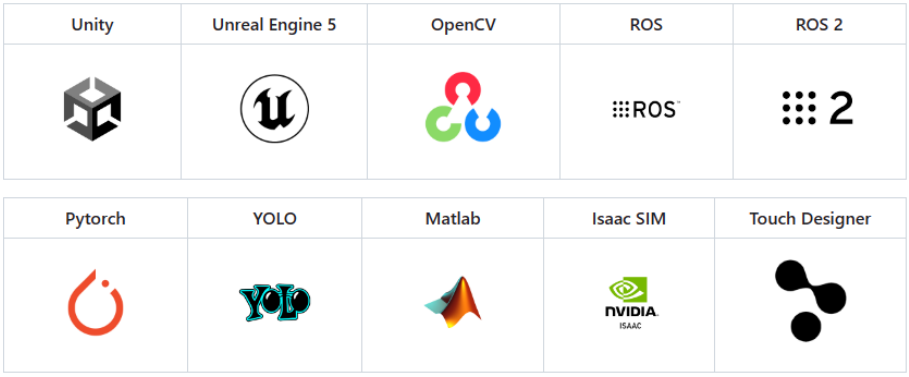

FIGURE 2.3 – APIs used in ZED

not move significantly. In the case of a video, it tracks object movements and precisely distinguishes each object, providing significant added value for our glasses.

2.2.3 2D Object Detection Model :

YOLACT :

The YOLACT model (You Only Look At Coefficients) is an object detection architecture distinguished by an innovative approach. Instead of directly predicting the coordinates of object bounding boxes, YOLACT focuses on predicting semantic masks and coefficients associated with bounding boxes. This approach allows for simultaneous object detection and semantic segmentation.

The YOLACT model structure consists of two key components : the mask generator and the bounding box detector. The mask generator takes an image as input and produces semantic masks for each detected object instance, defining specific spatial regions for each object.

The bounding box detector takes the input image and generates bounding boxes for each detected object. These bounding boxes are accompanied by coefficients that merge information from the generated masks with the bounding boxes.

An important point is that YOLACT is a real-time object detection model, making it well-suited for the project's objective. However, to fully leverage YOLACT.

2.2.4 MMYOLO :

Model Explanation :

The MMyolo model is built on the MMDetection framework and shares a similar code structure. It stands out as one of the most performant object

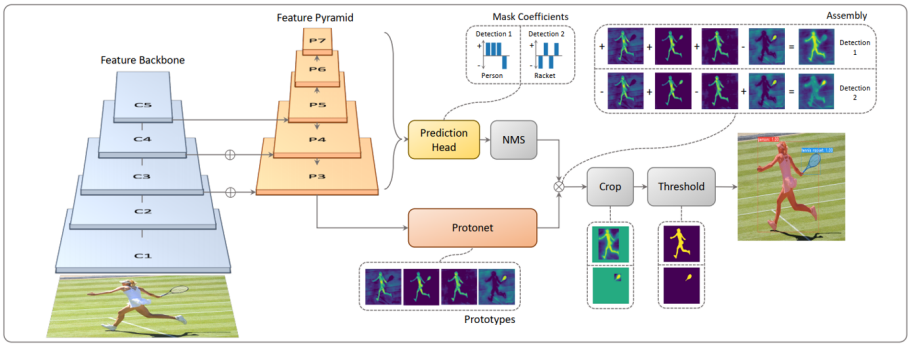

FIGURE 2.4 – Yolact Architecture

detection models to date, capable of operating in real-time with an impressive processing speed of 322 images per second.

Task	Dataset	AP	FPS(TRT FP16 BS1 3090)
Object Detection	COCO	52.8	322
Instance Segmentation	COCO	44.6	188
Rotated Object Detection	DOTA	78.9(single-scale)/81.3(multi-scale)	121

TABLE 2.1 – Performance of MMyolo Model

The MMyolo model, based on the YOLO series from OpenMMLab, adopts a convolutional neural network architecture for object detection. Trained on abundant annotated image data, it learns to identify various objects. Given an input image, the model generates predictions regarding the detected objects, including their class and position. The input parameters consist of the images to be processed, while the predictions of the detected objects constitute the output parameters of the MMyolo model.

It distinguishes itself through ease of customization and extension, characterized by a modular structure enabling users to merge different modules based on various training and testing strategies

MoroccoAI

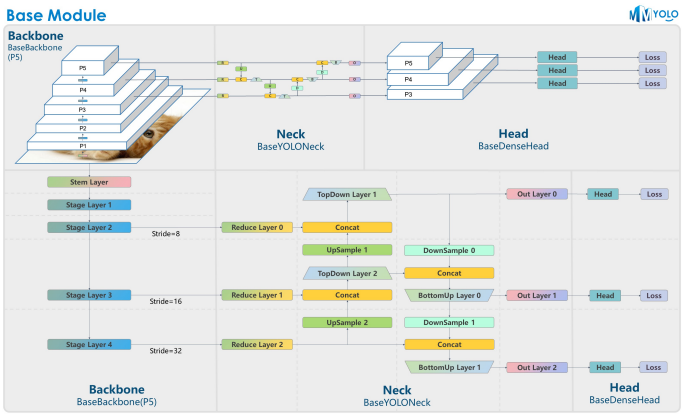

FIGURE 2.5 – MMyolo Architecture

Chapitre 3

Version 0.1

3.1 Presentation of the Model and Prompting Strategy

After understanding the overall vision of our project and the tools that can be used in this final version, such as the example of a 3D camera and real-time models, we move on to our first version, which will be a simpler example. In this version, the goal is to describe an image and have interaction between the model and the person. The objective is to arrive at a model that can perfectly describe the image—what exists in the image, their positions, whether they are close or not, whether they are to the right or left, etc.

It is important to note that the model at this stage is not a normal image-to-text model, as those models provide only a simple description of the image, which will not be useful for our work. The following diagram illustrates the different steps carried out by the model :

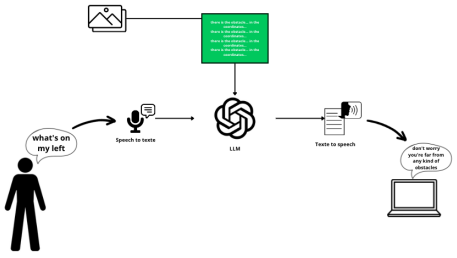

FIGURE 3.1 – VisionVista Simplified Model

3.1.1 Used Models :

In this section, we will discuss all the models used in this first version. It is important to note that the objective of this initial test was not to achieve a

model that responds in real-time but at least one that can provide a response. This is why we did not choose one of the real-time detection models, such as ZED 3D Object Detection, MMyolo, and YOLACT described in the previous chapter. Instead, we focused on the power and precision of the model in detection.

Detr-resnet-50

The DETR (End-to-End Object Detection) model with a ResNet-50 backbone is an end-to-end object detection model trained on the COCO 2017 object detection dataset (118k annotated images). It was introduced in the paper "End-to-End Object Detection with Transformers" by Carion et al. and first released in this repository.

Model description : The DETR model is an encoder-decoder transformer with a convolutional backbone. Two heads are added on top of the decoder outputs to perform object detection : a linear layer for class labels and an MLP (multi-layer perceptron) for bounding boxes. The model uses so-called "object queries" to detect objects in an image, with each object query searching for a specific object in the image. For COCO, the number of object queries is set to 100.

The model is trained using a "bipartite matching loss," where the predicted classes and bounding boxes of each of the N = 100 object queries are compared to ground truth annotations, padded up to the same length N. The Hungarian matching algorithm is employed to create an optimal one-to-one mapping between each of the N queries and each of the N annotations. Standard cross-entropy (for classes) and a linear combination of L1 and generalized IoU loss (for bounding boxes) are used to optimize the model parameters.

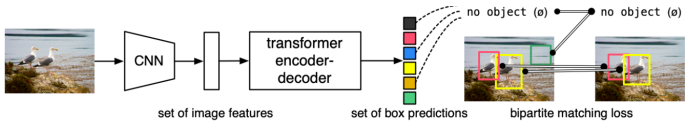

FIGURE 3.2 – Detr-resnet-50 architecture

GLPN-NYU

The GLPN-NYU model, a novel architecture for monocular depth estimation. Addressing the challenges of depth prediction from a single image, the model employs a hierarchical transformer as the encoder to capture global dependencies and multi-scale context features. In contrast to existing approaches, the GLPN-NYU incorporates a global-local path network, leveraging the benefits of transformers for enlarging the receptive field and skip connections for preserving short-distance information. The decoder utilizes a selective feature fusion module, which efficiently combines global and local features, achieving

superior performance with lower complexity. To further enhance the model, a depth-specific data augmentation technique, a variant of CutDepth, is proposed, considering the vertical position's significance in depth estimation. Experimental results on the NYU Depth V2 dataset demonstrate state-of-the-art performance, showcasing the model's generalization ability and robustness against image corruption.

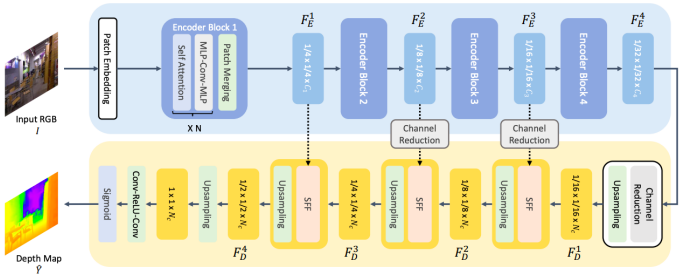

FIGURE 3.3 – GLPN-NYUA architecture

Speech-to-Text and Text-to-Speech :

In this section, we will use, for the prototype, the speech recognition model for converting voice to text and the pyttsx3 model for converting text to speech. Other models are available for this task, depending on the intended language and the quality of the models. However, in our case, we are opting for these models in the prototype.

3.1.2 Prompt engineering :

In this section, we will discuss the prompt techniques that we will use in our model. The first technique is called "one-shot prompt and few-shot prompts," simply involving providing examples with model-expected responses to help it understand the requested task. The second technique is "Chain of Thought prompting," which will assist us in structuring the actions we feed into the model and aid in obtaining accurate responses.

Zero-Shot Prompting :

Definition : Zero-shot prompting is a technique in which the model is prompted to perform a task for which it has not been explicitly trained, without providing specific examples for that task. This allows the model to generalize its learned knowledge from training data to accomplish a variety of tasks, even if it hasn't been exposed to those specific tasks.

Operation : When using zero-shot prompting, the model is presented with a description or a general directive for the task at hand without providing specific

MoroccoAI

examples. The model utilizes its prior knowledge to interpret and respond to the task appropriately, even if it differs from those encountered during training.

One-Shot Prompting :

Definition : One-shot prompting involves giving the model a single example of the task one wants it to perform. Unlike zero-shot, which provides no instances of the task, one-shot prompting offers a sole instance to assist the model in understanding the nature of the requested task.

Operation : By using one-shot prompting, the model examines an example of the task to be performed, usually in the form of a question-answer pair, and then generalizes this information to perform similar tasks when faced with similar prompts.

Few-Shot Prompting :

Definition : Few-shot prompting extends the concept of one-shot by providing a few examples (usually less than five) of the task one wants the model to accomplish. This gives the model a limited set of examples to understand and generalize the requested task.

Operation : With few-shot prompting, the model examines multiple examples of the task at hand. These additional examples enable the model to adjust its internal weights to better adapt to the specific task. Using a few examples helps provide more specific instructions without the need for large training datasets.

Chain of Thought Prompting :

Definition : Chain of Thought Prompting is a technique that involves structuring prompts in a sequential or logical manner to guide the model through a series of related steps. Instead of providing a single prompt, this method breaks down complex tasks into a sequence of interconnected prompts to guide the model's thought process.

Operation : When employing Chain of Thought Prompting, the user develops a series of prompts in a coherent order, where each prompt builds upon the information from the previous one. This sequential arrangement helps in guiding the model to generate more nuanced and contextually relevant responses. It serves as a method to direct the model's thinking process, allowing it to consider information in a step-by-step manner. Example Scenario :

Suppose the user, wearing AI-powered blind glasses that provide real-time video feedback of the surroundings, wants to interact with the environment using natural language commands. The Chain of Thought Prompting can be structured as follows :

Prompt 1 : Object Detection

User Query : "What objects are currently in front of me ?"

Model Guidance : Identify and locate objects in the live video feed using 3D object detection.

Prompt 2 : Spatial Arrangement

User Query : "How are these objects arranged ?"

Model Guidance : Describe the spatial relationships and positions of the identified objects in the scene.

Prompt 3 : Additional Details

User Query : "Tell me more about the bookshelf on my right."

Model Guidance : Provide additional details about specific objects, such as colors, textures, or any unique features, focusing on the bookshelf in this instance.

Prompt 4 : User Demands

User Query : "Did someone leave a bag on the table ?"

Model Guidance : Review previous video frames, using the glasses' memory to identify recent changes, and respond to the specific query about the bag on the table.

Prompt 5 : Summarize Scene

User Query : "Give me an overall summary of the current room."

Model Guidance : Summarize the key aspects of the scene, considering all previous inputs and queries, providing a comprehensive overview.

By using this chain of thought, the model is led through a logical sequence to generate a comprehensive description of the image, breaking down the complex task into manageable steps.

3.2 Tests :

In this section, we will outline all the tests conducted leading up to the final results.

3.2.1 Detection of objects in the image :

The initial tests were conducted in the computer vision domain. Our goal was to test various models and extract essential information so that the Language Model (LLM) could comprehend the content of the image. We focused on three pieces of data for the object detection model : the name of the detected object's class, the certainty of the detection, and the bounding box data. Certainty indicates how confident the model is in the detection ; the closer this parameter is to 1, the more aware the model should be of its presence, and the farther from 1, the more uncertain the model is. The bounding box coordinates help us detect two different pieces of information : the first is the relative position of the object to the image (left or right, top or bottom) by dividing the center of the object over the dimensions of the object

$$\left(\frac{\mathrm{x_{min}} + \mathrm{x_{max}}}{2X}, \frac{\mathrm{y_{min}} + \mathrm{y_{max}}}{2Y}\right)$$

with (X, Y)being the dimensions of the image and $x_{\min}, x_{\max}, y_{\min}, y_{\max}$ being the coordinates of the bounding box. The second piece of information is the relative distance of the object from the camera. To obtain this information, we use the depth estimation model's depth matrix (which is the same size as the image as it provides the depth of each pixel) and focus on the coefficients between xmin, xmax, ymin, ymax to extract the coefficient with the smallest value (as the distance from a point to a space of points, by definition, is the minimal distance). The following image illustrates the output obtained after all the modifications made. The following image shows the output obtained after all modifications were made :

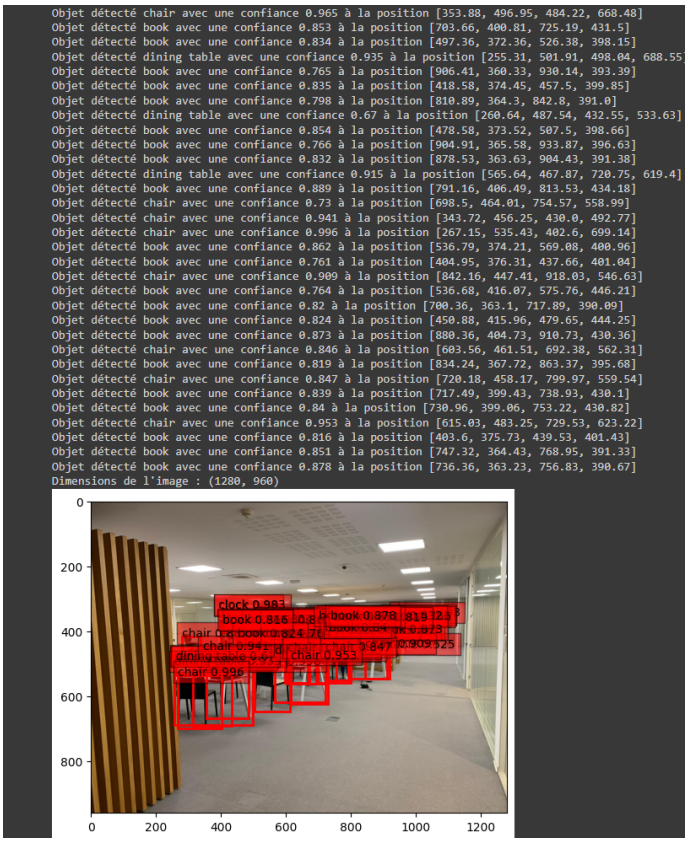

FIGURE 3.4 – Result of the object detection model

3.2.2 Estimation of depth in the image :

For the depth estimation part, the data of interest is the depth matrix. This matrix provides the relative position of each pixel in the image from the source (camera) on a scale from 0 to 10 meters using the GLPN-NYU model. If an object is beyond 10 meters, the model considers it as being at a distance of 10 meters, but this is not a significant issue at this stage, as it is far enough to be considered a potential hazard, which is sufficient for our initial test.

Now, to structure the information provided by the depth matrix, we will attempt to reduce it to an nn form, with n considered as a hyperparameter of the model. The choice of n divides the image space into nn, allowing us to know the distance of each region of the image by averaging the depth values of the pixels in each region. This technique facilitates the interpretability of the data in the matrix for the LLM model (it can distinguish directions and positions easily based on the approach of choosing the hyperparameter n) and respects the limitation of tokens that can be used for a model (generally, the size of the

FIGURE 3.5 – Depth estimation result

depth matrix is enormous and exceeds the tokenization limit). In our test, we set n to 8, and it was a good initial choice that yielded good results in the end.

3.2.3 Creation of the prompt :

Now that all the ingredients are ready, let's move on to creating the prompt. The plan is to craft an initiation prompt (one shot) that explains the model's role and how to interact with the delivered data, along with creating the prompt that will be entered at each iteration. The initiation prompt will be fed to the model before starting the conversation with the user, and the initial version was as follows :

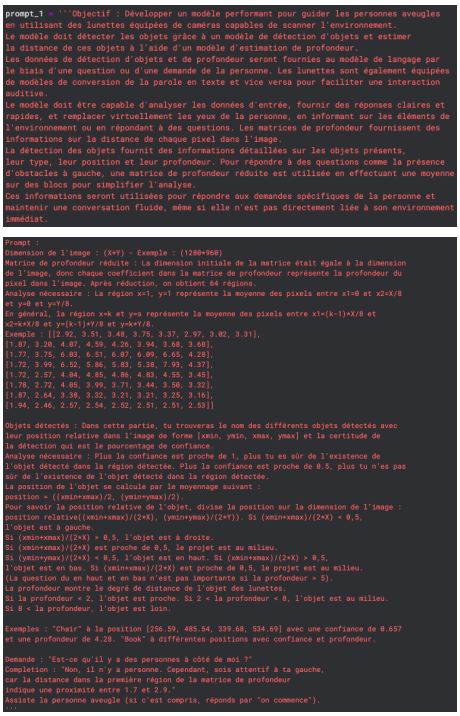

FIGURE 3.6 – initiation prompt

As soon as the user sends the image, we begin to accompany their request with the provided data from the computer vision part, structuring it in the following form :

```
def propmpt(size,summary_text,predicted_depth, talk):
     return f'''image size :{size}
     Simplified matrices of the depth :{predicted_depth}
    The detected objects and their positions :{summary_text}
    Request :{talk}'''
```

FIGURE 3.7 – iterative prompt

The "size" parameter represents the dimension of the image, the "predicted_depth" parameter represents the reduced depth dimension, the "summary_text" parameter represents the detected objects with their certainty, position, and depth, and the "Talk" parameter represents the person's request.

3.2.4 LLM Choice :

We chosed 3 models : GPT-3.5, Bard, and Gemini. Tests were conducted on all three, and the results were quite satisfactory for all cases, with minor diffe-

MoroccoAI

rences. Bard and Gemini proved to be quicker at understanding our initiation prompt than GPT-3.5. They respond directly by starting to assist the person, as described in the initiation prompt.

Bard and Gemini provide free APIs, while GPT-3.5 only grants free access for the first 30 requests, becoming paid thereafter. Bard's API sometimes gets blocked and requires periodic changes to continue.

Responses to user requests are a bit lengthy for all 3 models and require condensation to be useful in a real-world scenario.

For the 0.1 Version, we used Bard, and for the second ones, we opted for Gemini.

3.3 Results :

We relied on the Streamlit library to create interfaces, and here's an example of applying our model to an image. First, you open the model, and this interface appears :

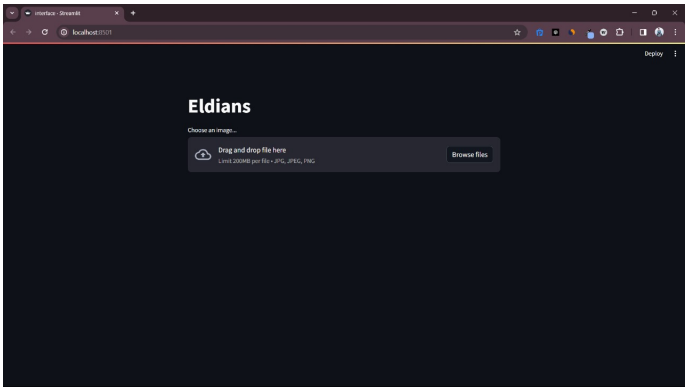

FIGURE 3.8 – Step 1

You click on "drag and drop file here" and choose the image you want to process. As soon as you select the image, it appears, and the image processing part begins to perform the analyses described in the "Tests" section.

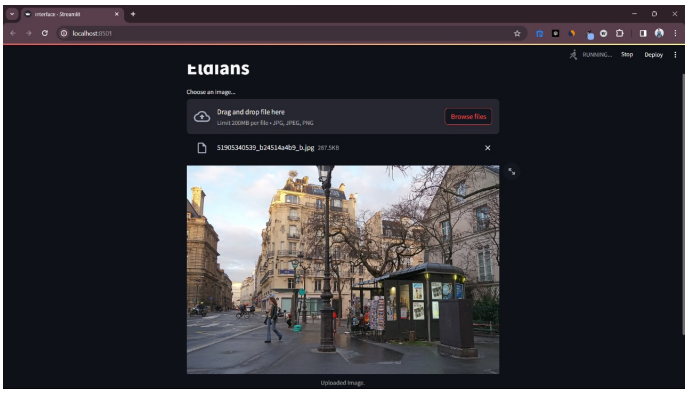

FIGURE 3.9 – Step 2

When the model is ready to answer your questions, it will respond with "ready", and a button named "start conversation" will appear. You click on the

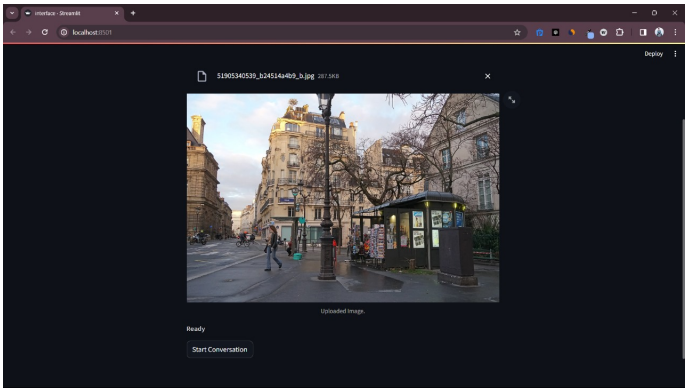

FIGURE 3.10 – Step 3

button, and when you see the phrase "Listening for speech input...", you can start speaking and ask anything you want about the image. If you want to end the conversation, you can click on "Stop conversation".

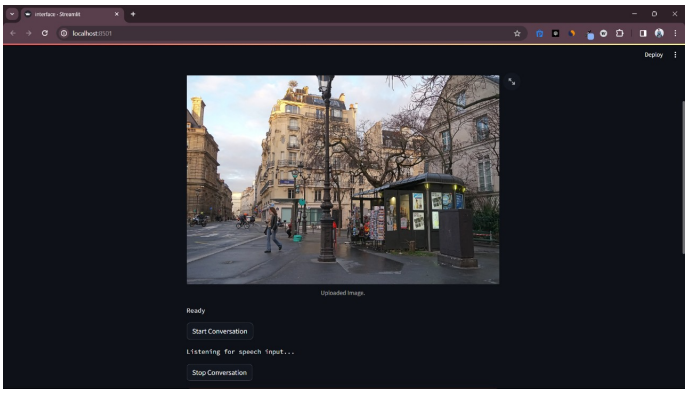

FIGURE 3.11 – Step 4

Your dialogue with the model will be displayed on the screen if you want to keep a record of your conversation.

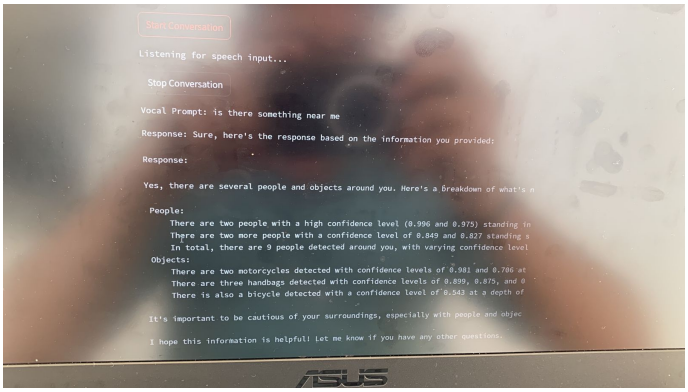

FIGURE 3.12 – Step 5

3.4 Limitations of version 0.1 :

Among the issues we noticed in this initial version, the primary concern is the execution time. Between the integration of the image and the first response, it takes about a minute, which may be acceptable if the goal is simply to interact with the model for image description. However, this is not suitable for the project's final objective of assisting visually impaired individuals. One of the causes of this delay is related to the models used in the image processing part. These models are not real-time detection models, and we use two separate models due to the image dimension (2D and not 3D). One solution is to use one of the models described in the "inspirations" section, either in 2D or 3D, which will speed up the process and reduce response time.

MoroccoAI

Another problem is in the speech-to-text part, which is too sensitive to the user's environmental noise. The speech-recognition model is not optimal for addressing this issue, even with modifications to its architecture. That's why we need to look for another model capable of eliminating noise and quickly capturing the user's request.

A third issue concerns the nature of the model's responses. It doesn't provide direct and concise answers but rather complete texts. Although it describes the image in detail, this is unnecessary because the goal is to alert the person in case of danger and respond to their request. Therefore, we need to modify the prompt to improve the quality of the responses.

Another limitation is the continuity of the conversation ; the model struggles to recall the history with the user, leading it to lose track and provide random responses, resulting in a discontinuity of actions.

Chapitre 4

Version 0.2

4.1 Presentation of the Model and Prompting Strategy :

In this version, the overall structure of the model remains the same. The only change is the addition of a memory to the model, meaning it will be capable of retaining previous questions and the context of the conversation. We also altered the initial prompt and iterative prompt after several tests, and switched the large language model to Gemini-pro.

4.2 Used Models :

We kept the same models for the image processing part, with a single modification to standardize the image size. After several tests, we noticed that the size plays a role in processing speed, so we set it to (512,512). We also kept the same models for the audio part, with the only change being at the linguistic model level.

Gemini-pro :

Gemini-pro is a variant of the Yahoo ! Gemini family of generative AI models. This model is designed to handle both text input and output. Its latest update is from December 2023, and its model code is "models/gemini-pro". Gemini-pro balances capacity and efficiency, offering a model size that ensures optimal performance. It is suitable for generating text, can handle multi-turn conversational formats, as well as "zero", "one", and "few-shot" tasks. Compatible generation methods include "generateContent", with a limit of 30,720 input tokens and 2,048 output tokens. The model is secured with automatically applied security parameters, adjustable by developers. The throughput limit is 60 requests per minute.

Variation	Attribut	Description
Gémeaux Pro	Dernière mise à jour du modèle	Décembre 2023
	Code du modèle	`models/gemini-pro`
	Capacités du modèle	• Entrée: texte • Sortie: text • Génère du texte. • Peut gérer le format conversationnel multitours. • Peut gérer des tâches "zéro", "un" et "peu de plans".
	Méthodes de génération compatibles	`generateContent`
	Limite de jetons d'entrée	30 720
	Limite de jetons de sortie	2 048
	Sécurité du modèle	Paramètres de sécurité appliqués automatiquement qui peuvent être ajustés par les développeurs. Pour en savoir plus, consultez la rubrique Paramètres de sécurité.
	Limite de débit	60 requêtes par minute

FIGURE 4.1 – Gemini-pro overview

4.3 Tests :

4.3.1 Detection of objects in the image :

In this version, we have changed the data structure provided by the object detection model up to the iterative prompt. Instead of simply providing the bounding box coordinates and letting the model perform its analyses, we have opted to calculate all the necessary values and provide them directly to the model, leaving it only the interpretation task. The idea is to provide the distance from the object's barycenter in the R^3 space in the form (x_g, y_g, z_g) with $x_g = \frac{xmin+xmax-X/2}{2}$ and $y_g = \frac{ymin+ymax-Y/2}{2}$, and z_g represents the depth of the object calculated in the same way as in the first version. By adopting this approach, we aim to achieve smoother responses. The user's (or glasses) position will be set as the point with coordinates (0,0,0). This way, the model can comprehend the distances and relative positions of objects from the user.

4.3.2 Prompt enhancement :

Now, with the change in data from the image processing part, we have also adjusted the prompt, but it's not just a simple change in the structure of data from the image processing part. Through our initial tests, we observed several issues in response quality and attempted to align the prompt to avoid them and better guide the model.

The iteration prompt was also modified to remind the model, in each iteration, of its role and how it should behave and respond. We noticed that this action is crucial because after a while, the model starts to forget the initiation prompt a bit (it begins to diverge a bit, but it still produces good results), so with this method, it always maintains its role.

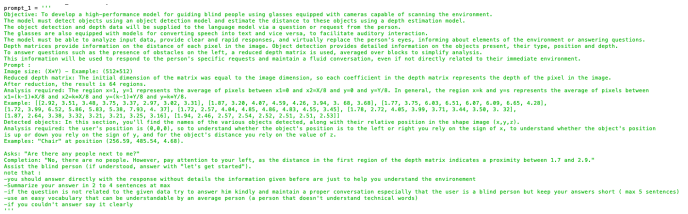

FIGURE 4.2 – Prompt d'initiation

```
def propmpt(size,summary_text,predicted_depth, talk):
    return f'''image size :{size} Simplified matrices of the depth :{predicted_depth} The
    note that :
-you should answer directly with the response without details the information given before
-Summarize your answer in 2 to 4 sentences at max
-if the question is not related to the given data try to answer him kindly and maintain a p
-use an easy vocabulary that can be understandable by an average person (a person that does
-if you couldn't answer say it clearly
-give the answer directly(as you are an assistant to a blind person)'''
```

FIGURE 4.3 – Prompt itérative

4.4 Results :

In this version, we've successfully resolved several issues. The initial problem was associated with the context and length of responses. The 0.1 version of the model struggled to deliver clear, direct, and concise answers. However, by introducing an analytical approach between the output of the image processing stage and the prompt, coupled with prompt modifications, we've significantly improved the model's performance. It can now provide appropriate responses to questions in just a few words.

Another challenge addressed pertains to the flow of the conversation. In the first iteration, each question operated independently, leading to difficulties in contextual understanding when transitioning between images. Now, with the integration of Gemini-Pro and various code adjustments, the model can sustain a coherent dialogue with the user. Even if presented with a question unrelated to the image, it responds appropriately (e.g., showing empathy towards a visually impaired user). Subsequently asking an image-related question results in a response that considers the contextual background.

When an image is changed, and a question is posed, the model not only responds but also establishes connections between the contexts of both images (previous and current) if the user clarifies the context and relationship between them.

The speed of the image processing stage has been significantly enhanced by implementing the concept of reducing image dimensions. In the initial version, this process took between 1 minute and 1 minute 20 seconds. Presently, it completes in approximately twenty seconds.

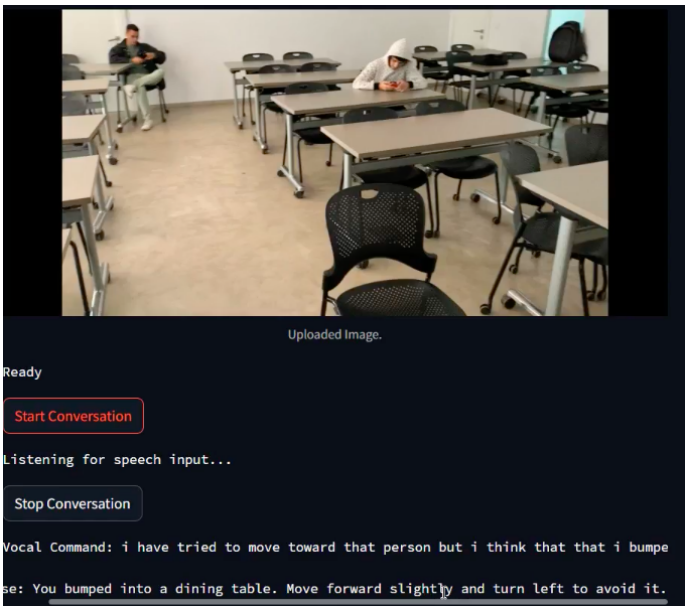

FIGURE 4.4 – Conversation example

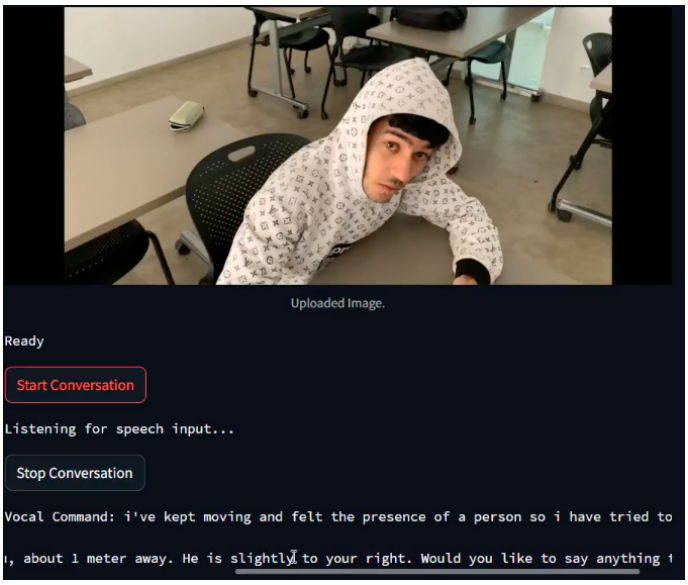

FIGURE 4.5 – Changing the image

MoroccoAI

4.5 Limitations of Version 0.2 :

Version 0.2 shows promise, particularly in its ability to respond to questions fluently and with good contextualization. However, several unresolved issues have been identified.

Firstly, there are limitations in terms of image processing. The models do not operate in real-time, hindering the deployment of the project for real-time applications, such as live use. Additionally, the object detection model is confined to a limited number of classes on which it was trained. Significant work is required in this area to align the model with our project. Understanding the daily behavior of a blind person is essential to derive interactions and common "objects," enabling the definition of either a new model with identified object classes or the application of transfer learning to a real-time model.

Regarding the "prompt" aspect, there is still room for development. Fine-tuning the model may be considered to enhance its performance.

As of now, our model only operates with images, which may be inconvenient for users who must capture an image each time they want to query the model. Additionally, the model struggles to read text within images, a critical feature for individuals seeking directions or information about their surroundings.

Chapitre 5

Conclusion and Perspectives

In conclusion, Version 0.2 of our intelligent glasses project, developed during the Morocco AI 2023 hackathon, demonstrates significant promise in addressing the challenges faced by individuals with visual impairments. The ability to respond to questions fluidly and in context is a notable achievement. However, as we move forward, it is essential to acknowledge and address persistent issues that hinder the seamless integration and functionality of the system.

The foremost limitation lies in the realm of image processing. The current models lack real-time capabilities, impeding the deployment of the project for live applications. Additionally, the object detection model is constrained by a limited set of trained classes, necessitating an in-depth exploration of the daily interactions of visually impaired individuals to expand and refine the model. This may involve defining new object classes or implementing transfer learning on a real-time model.

The "prompt" functionality also presents opportunities for further development. Considering fine-tuning for ongoing improvement could enhance the model's responsiveness and effectiveness in assisting users.

At present, the model operates exclusively with images, presenting a constraint for users who must capture an image each time they wish to query the system. Furthermore, the model struggles to interpret text within images, a critical feature for users seeking directional guidance or information about their surroundings.

Moving forward, our focus will be on refining image processing capabilities, expanding object detection classes, and exploring opportunities for prompt-based fine-tuning. Additionally, the integration of real-time functionality and text interpretation within images will be pivotal for enhancing user experience and providing comprehensive assistance to individuals with visual impairments. We remain committed to advancing visual assistance technologies and hope that our efforts inspire further benevolent initiatives in the field of AI.