

*Artificial Intelligence and Data Engineering*

*Industrial* *Applications*

***Car Suggestion***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prototype Documentation

Luca Caprioli

Martina Marino

Roberta Matrella

Academic Year 2021/2022

Summary

[Introduction 3](#_Toc131965137)

[Goal of the project 3](#_Toc131965138)

[System description 4](#_Toc131965139)

[Hardware components 4](#_Toc131965140)

[Software components 4](#_Toc131965141)

[Emotion recognition score 4](#_Toc131965142)

[Application functionalities 5](#_Toc131965143)

[System functioning diagram 6](#_Toc131965144)

[DeepFace 6](#_Toc131965145)

[Architecture 7](#_Toc131965146)

[User Manual 8](#_Toc131965147)

[Conclusion 14](#_Toc131965148)

[References 16](#_Toc131965149)

# 

# Introduction

Autonomous cars will have a significant impact on the next years life. They will not only benefit those who are unable to drive, but will also change the way we live our lives. Thanks to the autonomous car it will be possible to reduce the chances of injury due to accidents, to minimize the need for parking space, thanks to the “sent back home” function, simplify the labor of the logistic companies. One of the most exciting prospects of autonomous cars is the new opportunities for productivity that will emerge. People will be able to make use of the time they would normally spend driving on other activities. This will bring about a new era of car travel that will enable people to take full advantage of this newfound time.

Many of the world's leading car manufacturers, such as Tesla, Mercedes-Benz, Ford, and Renault, have invested heavily in autonomous driving technology. In 2022, global spending on self-driving cars amounted to around $126.19 billion, and this is forecast to increase to $1,808.44 billion by 2030[1].

Furthermore, the importance of the recognition of emotions in understanding people’s preferences is increasingly recognized. Indeed, it is a powerful tool to adapt the travel to the preferences of the passengers.

Surely self-driving cars will use emotion recognition to offer different kinds of services to adapt every aspect of the experience and help people to make every kind of choice.

# Goal of the project

The main goal of this project is to extract the percentage of positive emotions of a user during a guide experience thanks to an emotion recognition system.

The project offers a GUI that allows the selection of different guide simulations with distinct characteristics. Moreover, is also possible to interact with the system using vocal commands that permit starting a simulation and modifying its properties.

This outcome has been achieved through the collaboration of some key components including:

* Audio acquisition
* Speech recognition
* Video reproduction
* Emotion recognition

# System description

## Hardware components

For the implementation of the system the following hardware have been used:

* Raspberry Pi 3 B+: tiny card, fast and versatile.
* Usb Microphone: connected to the Raspberry to receive vocal commands from the users.
* Pi Camera Module V2.1: camera connected to the Raspberry to recognize the emotions of the users during the simulations.

## Software components

A virtual environment has been created to run the application into the Raspberry Pi 3 platform to guarantee the isolation of libraries and Python interpreter. The version of Python used is 3.9.

* **Tkinter**: simple GUI library for Python. It allows to create rapidly an interface that works with Python.
* **SpeechRecognition:** library to perform speech recognition:
  + **Google Recognizer:** free and doesn’t require an API key to use it.
* **Tensorflow:** open-source artificial intelligence library used for the model on the server (pc).
* **PiCamera**: library used on Raspberry to take pictures and high-definition video.
* **Socket**: library that implements sockets to use for internet communication. It has been used to let the Raspberry communicate with the server on the pc.
* **Deepface**: it is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It has been used to recognize the emotion of the users during the simulation. It is a hybrid face recognition framework wrapping state-of-the-art models: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace and Dlib.

## Emotion recognition score

During the simulation the positive emotion recognized (happy, surprise) on the total of the emotions are counted. The main goal of the application is to understand how much the user liked the simulation and, for this reason, only the positive emotions have been considered. Given the number of frames in which positive emotions have been recognized and the total number of frames, the score is computed in this way:

Basically, the total number of positive emotions has been divided by the total. This score (as percentage) is shown to the user in the GUI.

## Application functionalities

At the start there is a panel that allows to specify the details of the simulation the user wants. It is also possible to ask for a specific type of simulation using the vocal commands. After starting the application, the speech recognition is always active waiting for a new command. If the form provided by the GUI is compiled or a specific vocal command is used, the simulation starts. The simulation consists of a video that changes based on the place. In the GUI is also shown the speed, the scent and the temperature. Different perfumes imply a different interface color. It is possible, while the video is playing, to change some simulation characteristics using vocal commands.

It is possible to use the following commands:

* **Take me to the**: reproduces a video where the place is the one specified after the command (es. sea, mountain…)
* **Now to the**:allows to change the scenario of the simulation (change of the video in playing)
* **Slower**: reduces the speed, the video playing is slower
* **Faster**: higher speed video
* **Perfume**: changes the scent (simulated through the GUI color)
* **Temperature**: change the temperature
* **Stop**: close the simulation

The commands recognized and the emotion score performed during the simulation are shown in a specific panel at the bottom of the GUI.

The frames collected by the Raspberry camera are sent to the server hosted on the pc that elaborates the data and returns the score to the application on the Raspberry.

## System functioning diagram

The following schema represents the interaction between the different parts of the system:

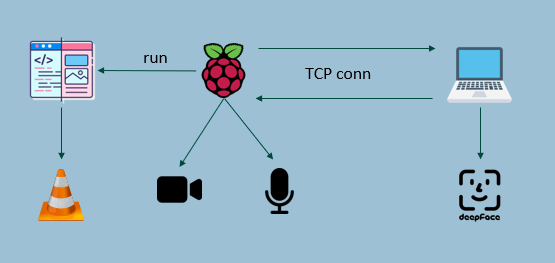


Figure 1 System schema

The Raspberry must be connected to the network to make use of the services offered by Google. Moreover, the user can interact with a user interface.

The structure of the prototype is client-server. To achieve a better result in terms of time, the network to evaluate the emotion is hosted on a server, in this case a pc, that uses the DeepFace network. At the end of the video the score of the emotions is shown in the GUI.

On the client side, thanks to the microphone, the vocal command of the user is recognized. Based on the command, different videos and GUI elements are displayed. During the simulation experience, the camera records the images of the user face that are sent to the server on a TCP connection.

On the server side, the images received by the client are elaborated, thanks to the DeepFace network, which associates to each image the most dominant emotion and, when the end of the simulation is notified by the client, the score is sent back.

## DeepFace

DeepFace [2] is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It is a hybrid face recognition framework wrapping state-of-the-art models: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepID, ArcFace, Dlib.

It was [trained](https://en.wikipedia.org/wiki/Machine_learning) on 4 million images uploaded by Facebook users. The Facebook Research team has asserted that the DeepFace method reaches an accuracy of 97.35% ± 0.25% on Labeled Faces in the Wild (LFW) dataset.

By default, DeepFace returns the score for each of the seven emotions recognized but also the dominant emotion between all.

### Architecture

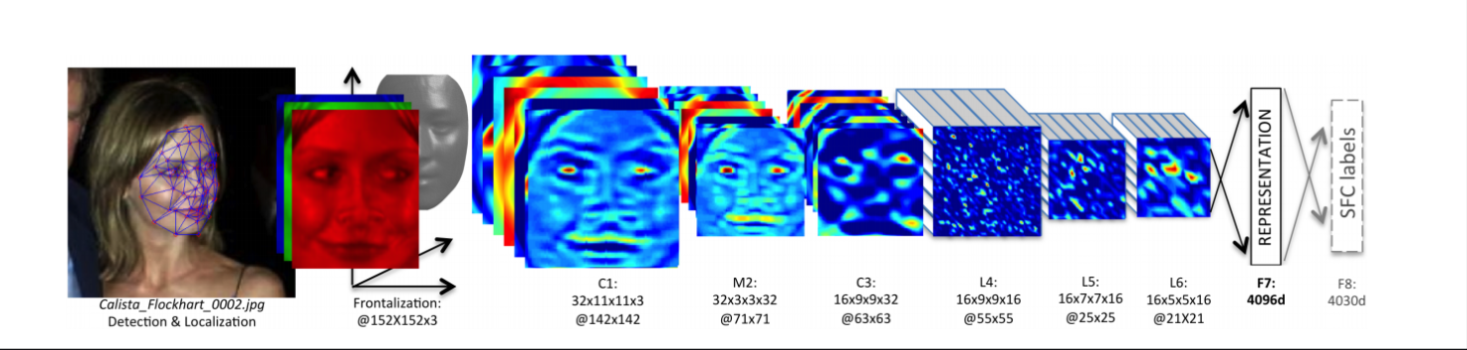
The DeepFace [2] system consists of four different modules: 2D alignment, 3D alignment, frontalization, and neural network. An image of a face is passed through them sequentially. The result is a 4096-dimensional [feature vector](https://en.wikipedia.org/wiki/Feature_(machine_learning)) that represents the face. The feature vector is then further processed for many different tasks. DeepFace uses fiducial point detectors based on existing databases to direct the alignment of the faces. DeepFace can be used to recognize a face, using the cosine similarity between feature vectors, but also to achieve emotion recognition.

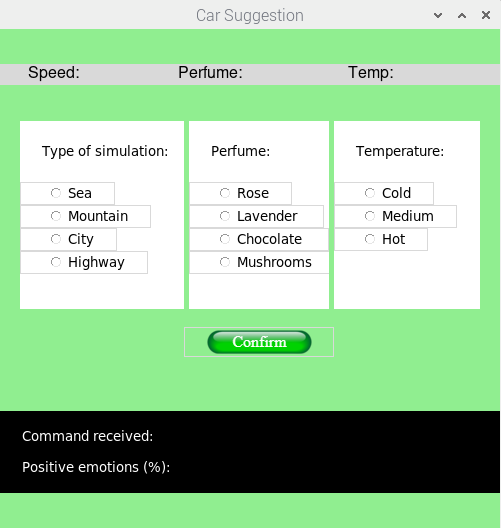
Figure 2 DeepFace architecture

It takes as input a 3D-aligned RGB image of 152x152 that is passed to the convolution layer with 32 filters and size 11\*11\*3 and a 3\*3 max-pooling layer with stride of 2. This is followed by another convolution layer of 16 filters of 9\*9\*16. The purpose of these layers is to extract low-level features from the image edges and textures. The next three layers are locally connected layers that give the model different discrimination ability. The last two layers of the model are fully connected layers, they help in establishing a correlation between two distinct parts of the face. The output of the second last fully connected layer is used as a face representation and the output of the last layer is the softmax layer K classes for the classification of the face.

# User Manual

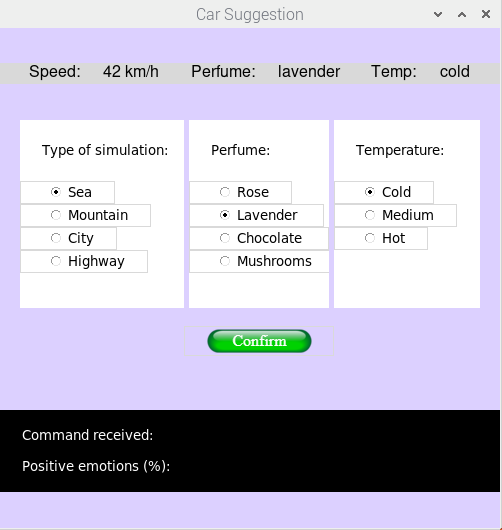
Starting the application, a GUI composed by 3 main sections is presented to the user:

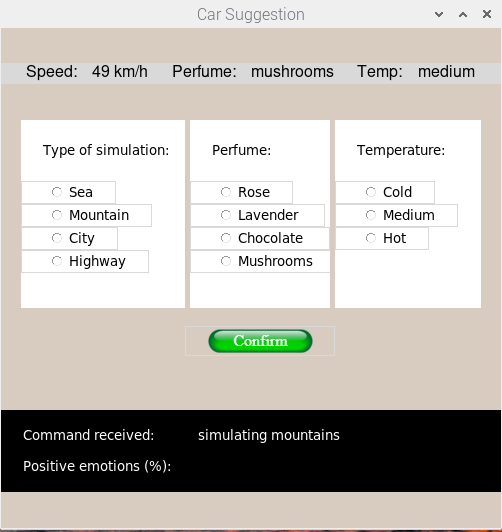
1. On top there are the current parameters of the simulation (empty at the start of the application).
2. In the middle there are the different options that can be selected to start manually a simulation.
3. On the bottom there are the vocal commands of the user detected by the VUI and the rate of positive emotions at the end of a simulation.



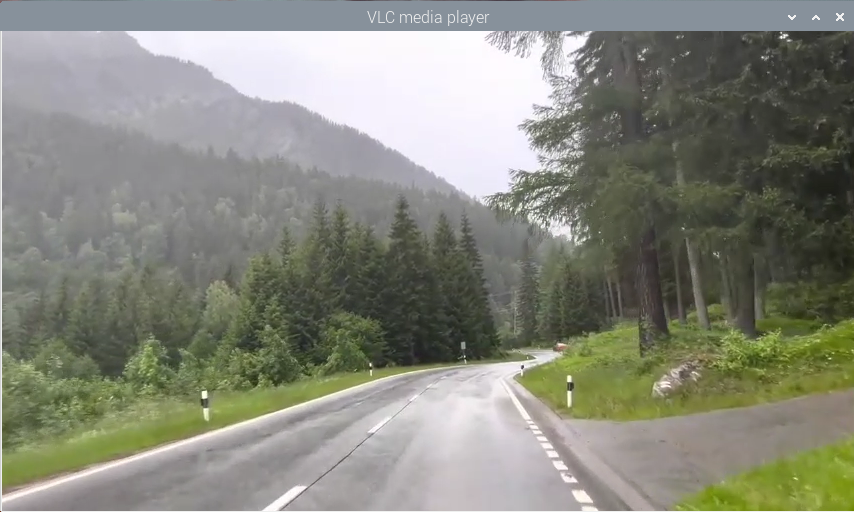
In the background there is a backlog running notifying the continuous wait of the VUI for vocal commands.

The user can either start a simulation manually by selecting the 3 different options and pressing the “Confirm” button or by issuing the “Take me to the [place]” vocal command. In the first case the simulation will start without any notification, in the second case the system will display on the GUI the vocal command detected.

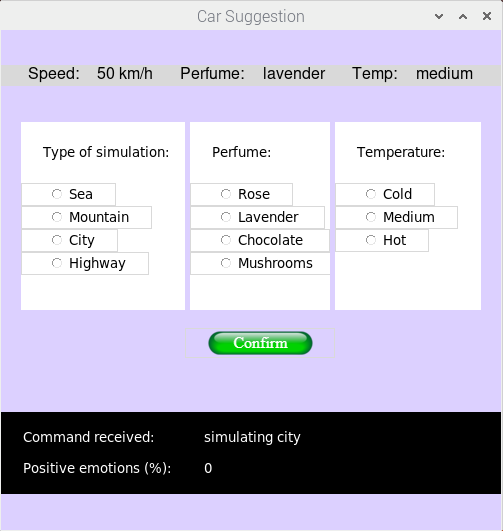




The simulation mockup consists in a driving video recorded in one of the environments presented in the GUI. Three different simulations for each environment are available.



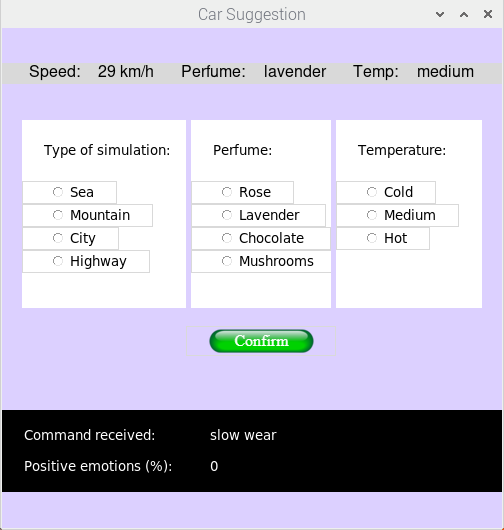
During the simulation, the user can change the scenario by issuing the vocal command “Now to the [option]”. In this way only the scenario changes and the user can use the vocal commands to customise the other parameters of the simulation.



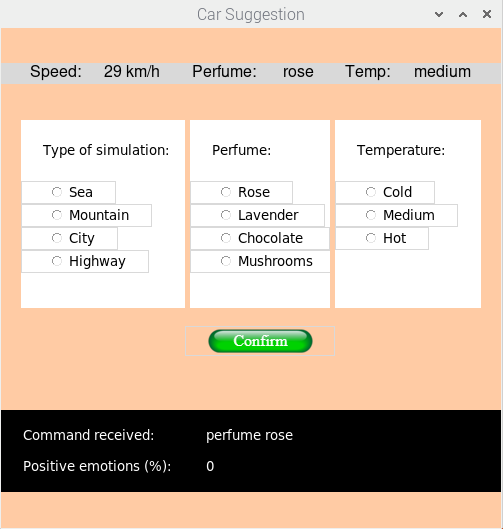


During the simulation, throught vocal commands, the user can modify the Speed, Perfume and Temperature parameters of the simulation throught the specified commands:

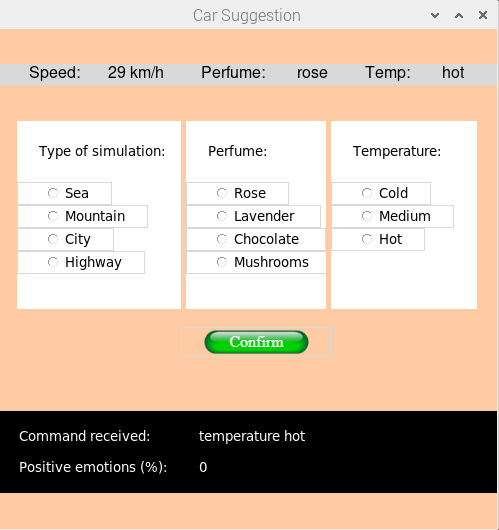
* Faster/Slower: the video speed is increased/decreased by double/half of the current value and a mockup speed in printed on the GUI.



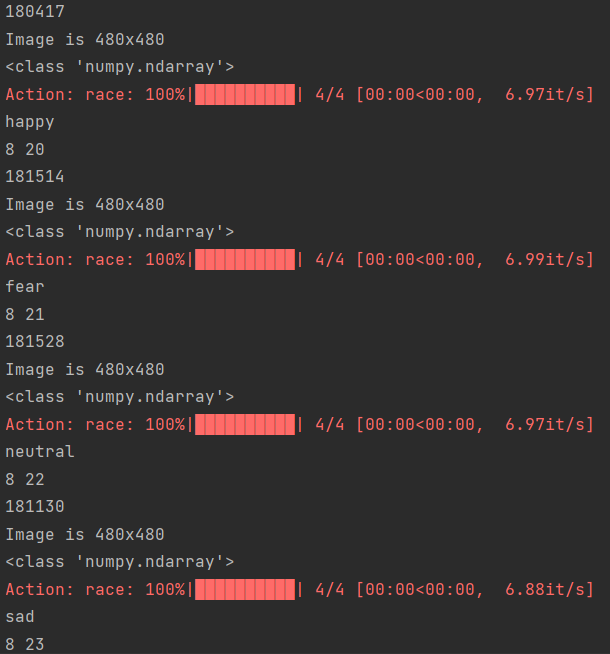
* Perfume [option]: the color of the GUI changes basing on the option specified and the option is printed.



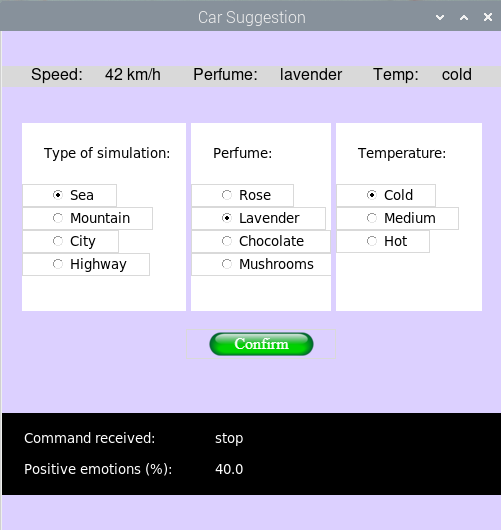
* Temperature [option]: the option is printed on the GUI.



During all the simulation, a connection is instantiated by the application with a server in charge of processing the snapshots sent by the camera to execute emotion recognition.



With the “Stop” command any current running simulation is halted, and the server is notified to send the percentile score of positive emotions detected during the experience (computed as described before). The result is then printed on the GUI.



# Conclusion

The project aims to develop a prototype of an autonomous car guide that integrates emotion recognition to offer personalized services to passengers. The system is composed of both hardware and software components, including a Raspberry Pi, a microphone, a Pi camera, and software libraries such as Tkinter, SpeechRecognition, Tensorflow, PiCamera, Socket, and DeepFace. The system operates through a client-server architecture, where the Raspberry Pi functions as the client and the server is hosted on a PC. The system provides a user interface composed of different panels that offer information on the simulation and enable the selection of different options to customize the experience. Through vocal commands, the user can modify some of the simulation parameters such as the scenario, the speed, the perfume, and the temperature. During the simulation, emotional recognition is performed through facial recognition analysis and the results are used to generate a positive emotion score that is displayed at the end of the experience.

The prototype represents a starting point to explore the potential applications of autonomous cars with emotion recognition capabilities. Future developments could include the integration of more advanced AI models and the implementation of other features for improved interaction between passengers and the autonomous car guide.

Code

The code of the project application can be downloaded on the following github link:

https://github.com/martimarino/Car-Suggestion

# References

[1] https://www.precedenceresearch.com/autonomous-vehicle-market

[2] https://ieeexplore.ieee.org/document/6909616