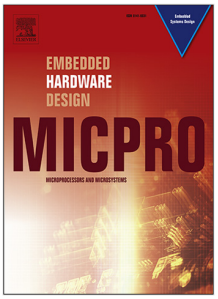
Journal Pre-proof



Basic human–robot interaction system running on an embedded platform

Julio Vega

PII: S0141-9331(21)00476-2
DOI: https://doi.org/10.1016/j.micpro.2021.104316
Reference: MICPRO 104316

To appear in: *Microprocessors and Microsystems*

Received date : 16 December 2020
Revised date : 3 June 2021
Accepted date : 14 July 2021

Please cite this article as: J. Vega, Basic human–robot interaction system running on an embedded platform, *Microprocessors and Microsystems* (2021), doi: https://doi.org/10.1016/j.micpro.2021.104316.

Basic human-robot interaction system running on an embedded platform

Julio Vega*

June 3, 2021

Abstract

Robotics will be a dominant area in society throughout future generations. Its presence is currently increasing in most daily life settings, with devices and mechanisms that facilitate the accomplishment of diverse tasks, as well as in work scenarios, where machines perform more and more jobs. This increase in the presence of autonomous robotic systems in society is due to their great efficiency and security compared to human capacity, which is thanks mainly to the enormous precision of their sensor and actuator systems. Among these, vision sensors are of the utmost importance. Humans and many animals naturally enjoy powerful perception systems, but, in robotics, this constitutes a constant line of research.In addition to having a high capacity for reasoning and decision-making, these robots incorporate important advances in their perceptual systems, allowing them to interact effectively in the working environments of this new industrial revolution. Drawing on the most basic interaction between humans, looking at the face, an innovative system is presented in this paper, which was developed for an autonomous and DIY robot. This system is composed of three modules. First, the face detection component, which detects human faces in the current image. Second, the scene representation algorithm, which offers a wider field of view than that of the single camera used, mounted on a servo-pan unit. Third, the active memory component, which was designed and implemented according to two competing dynamics: life and salience. The algorithm intelligently moves the servo-pan unit with the aim of finding new faces, follow existing ones and forgetting those that no longer appear on the scene. The system was developed and validated using a low-cost platform based on a Raspberry Pi3 board.

Keywords: Python; Low-cost; Raspberry Pi; Visual attention; Face tracking; Human-Robot interaction

1 Introduction

Over the last decade, technology has become increasingly common in most settings in daily and industrial life. In homes, there has been a significant increase in the presence of technological devices, such as computers, tablets, smartphones, domotic systems, etc. They are all interconnected through the Internet. At industrial level, rather than autonomous machines, factories are increasingly incorporating intelligent robots with sophisticated sensory systems into their production chains, with vision as the main mechanism of perception.

*Universidad Rey Juan Carlos, julio.vega@urjc.es

Figure 1: iRobot Roomba and Jet Braava, and domotic application Wattio

On the one hand, a home now often has numerous technological elements. The appearance of robotic devices in the mass market, such as robotic vacuum cleaners and mops (Figures 1 left and middle), as well as numerous applications and existing domotic services (Figure 1 right), has made this technology increasingly present in the daily routine of society. Our daily lives include many other frequently automated tasks, such as withdrawing money at the ATM, automatic payment in supermarkets, or the massive use of Internet, shopping, banking, etc.

On the other hand, the so-called *Industrialization 4.0* involves the integration of complex robotic systems in factories (Figure 2), logistics and what is known as the *Internet of things*, where sophisticated automatons handle an immense quantity of data to take strategic decisions for companies. The short and mid-term future is/will be marked by industrial production dominated by intelligent machines. The presence of humans in these *intelligent factories* will tend to be increasingly reduced, eventually being nominal and sporadic. There is no doubt that a machine's capacity for taking optimum decisions in real time and simultaneously handling an enormous quantity of data, is far greater than that of a human being.

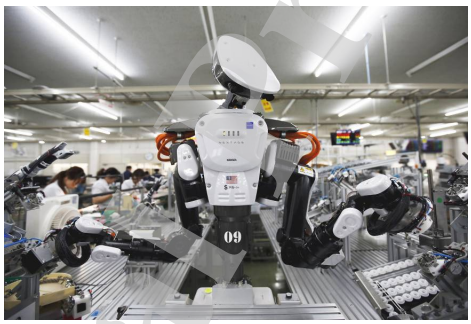


Figure 2: Intelligent robots at Glory Ltd

In addition, the aim of integrating a robot as much as possible in a real environment requires it to understand the natural forms of human communication. These robots are called *social robots*. The most basic, primitive form of interaction between humans is to look at a face [Ben Amor et al., 2014, Ewerton et al., 2015]. The face provides information not only about the position of the person interacting with the robot, but also about the identification of this person. Furthermore, it reflects the human's mood, and even their intention. Specifically, the most important part of the face, which transmits the most feelings, is the eyes [Faraj et al., 2020].

It is therefore interesting to develop techniques that allow the robot to know the position of the people around it and to follow them at all times [Bakar and Mohamad Amran, 2015]. Hence, for an acceptable robot-human interaction, a face detection and tracking mechanism [Parks et al., 2014, Menéndez et al., 2013] is indispensable. This allows a more fluid interaction with people, because they perceive the robotic interlocutor as more natural [Rautaray and Agrawal, 2015]. These mobile and intelligent robots need, in addition to a large computational capacity, a complex sensory

system to act intelligently not only in factories but in robot-human interaction at general level [Newcomb et al., 2015]. The fixed automation of structured production chains is giving way to an unpredictable world and a totally unstructured reality which makes evident the need for a wide complementary range of sensors and actuators to attain complete autonomy [Ci and Huang, 2016, Li et al., 2016].

Of these sensors, vision systems are currently one of the most widely used sensory elements in autonomous robotics. Their main difficulty lies in extracting useful information from the captured images, as well as the small visual field of conventional cameras. However, *active cameras* enable characteristics of a previously visited area to be revisited, even if such an area is out of immediate visual range. In order to obtain accurate information about the areas of interest around the robot, a detailed memory map of the environment is necessary. Since the computational cost of maintaining such an amount of information is high, only a few references can be maintained.

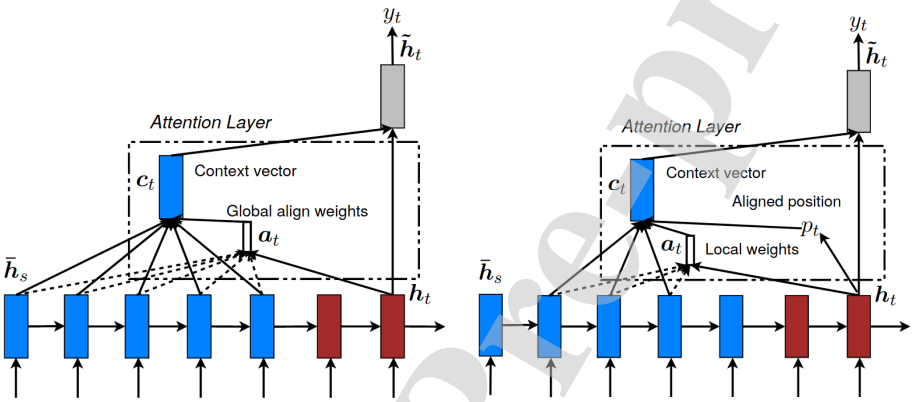


Figure 3: Global (left) and local (right) attention models

In autonomous robotics, it is important to perform visual attention control [Nobre, 2015]. The robots' cameras provide an extensive flow of data from which it is necessary to select what is interesting and ignore what is not. This is is known as selective visual attention. There are two aspects of visual attention [Luong et al., 2015]: the global (*overt attention*) and the local (*covert attention*). Local attention (Figure 3 right) consists of selecting, within an image, the data that interest us. Global attention (Figure 3 left) consists of selecting the objects that interest us from the environment around the robot, beyond the current visual field, and directing the gaze towards them.

Humans already naturally have a precise active vision system [Robinson, 1964, Clark and Stark, 1975, Bajcsy et al., 2016], which means that we can concentrate on certain regions of interest in the scene around us, thanks to the movement of our eyes and/or head, or simply by distributing gaze in different zones within the current image that are being perceived ([Biswas, 2016]).

This work presents a novel visual perceptive system, which was developed for a low-cost robot composed of two modules. The first is a short-term dynamic visual memory of robot surroundings. It takes images from a mobile camera, detects human faces and offers a wider field of view and greater robustness to occlusions than instantaneous images. This memory stores 3D points representing the human faces around robot. The memory contents are updated in a continuous coupling with the current image flow. Second, a gaze control algorithm was developed to select where the camera should look at each moment. It manages the movement of the camera to periodically re-

observe faces already stored in the visual memory, to explore the scene, and to test tentative face positions in a time-sharing fashion. Working in conjunction, these two modules build and update an attentive visual memory of the faces around the robot.

2 Visual attention system

The goal of the proposed system is to visually track the faces in the scene around the robot. It must detect new faces, track them, update their relative positions to the robot and remove them from the memory once they have disappeared. The perceptive system developed was designed for autonomous robots that use a single mobile camera, like that on the head of humanoids or in robots with pan-tilt units.

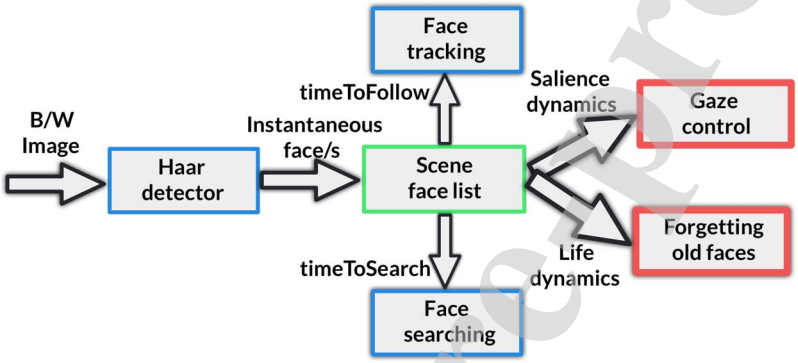


Figure 4: Block diagram of the proposed visual system

The different components on which the behavior of the developed system is based are shown in Figure 4 and are described below. The *Haar detector*, which is in charge of identifying the human faces in the current image, and the *Scene face list*, which composes the visual memory and allows the field of view to be extended to the entire scene around the robot, beyond the current field of view. In order to feed this visual memory, an overt attention algorithm was designed to continuously guide camera movements, choosing where to look at every moment. It was inserted inside this `active_visual_memory` component and associates two dynamic values with each face in memory: *salience* and *life* (quality). Faces with low life are discarded and faces with high salience are good candidates to look at.

2.1 Visual memory

The `active_visual_memory` component builds a short term visual memory of human faces in the robot's surroundings. The memory is built by analyzing each camera image looking for faces and updating its features already stored in the memory, such as their 3D position. The memory is dynamic and is continuously coupled with camera images. The new frames confirm or correct features of the human faces stored in memory, such as like their 3D relative position to the robot. New faces are incorporated into memory when they appear in images and do not match any known face.

This memory has a broader scope than the camera field of view and objects in memory have more persistence than the current image. Regular cameras typically have a 60-degree scope. This would be good enough for visual control but a broader scope may improve robot responses in tasks

like navigation, where the presence of obstacles in the robot's surroundings should be taken into account even if they lie outside the current field of view.

This memory is intended as local and short-term. Relative face positions are estimated using the robot's odometry. Being only short term and continuously correcting with new image data, there is little time to accumulate errors in the face's estimated relative position.

2.2 Scene representation

Starting from the detection of faces in the image perceived by the camera, the field of view of the system is extended, mounting the camera on a mechanical neck that allows it to be moved. Thus, it will capture images when the neck lands on some point. This movement of the mechanical neck is horizontal (pan), thus describing a kind of dome (Figure 5 left). If an image were taken with the on-board camera for each of the positions that the neck can take, there would be a large scene image composed of small monocular images (Figure 5 right).

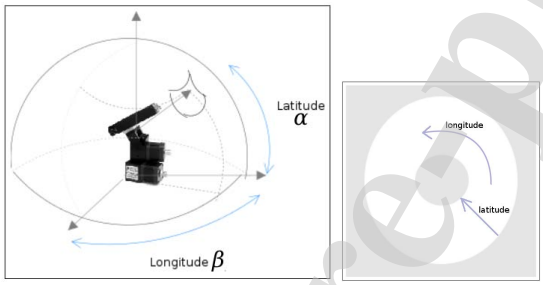


Figure 5: Neck movement scene (left) and resultant image from scene (right)

At this point, to form this scene image, it is necessary to make a correspondence between the pixels of the monocular image with the pixels of the scene image, since the monocular image is projected in the latter one. Depending on the position of the mechanical neck, the monocular image obtained by the camera has Cartesian coordinates (u, v) but the scene image also has coordinates (α, β) (Figure 5 right). To be able to build this scene image, each coordinate (u, v) of the monocular image (corresponding to each of its pixels) must be transformed to the coordinates (α, β) of the scene image.

It is known that, in the first column ($u = 0$) of the monocular image, the α value is $\alpha : pan + (\Delta\alpha/2)$, and its corresponding α value in the last column ($(u = umax)$) of the image is: $\alpha : pan - (\Delta\alpha/2)$, where pan is the pan value of the neck at that moment and $\Delta\alpha$ the value of the horizontal camera aperture, which is 53.50 ± 0.13 for the `PiCamera`[1] used in the platform developed. As a result, two points can be extracted from the line which transforms the image coordinates into the scene coordinates and vice versa: $P1(0, \Delta\alpha/2 + pan)$ and $P2(umax, pan - \Delta\alpha/2)$. The value u is replaced by the column and we have the value α. This equation is 1.

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) \Longrightarrow \alpha = \frac{P_2 \cdot y - P_1 \cdot y}{P_2 \cdot x - P_1 \cdot x}(u - P_1 \cdot x) + P_1 \cdot y$$

2.3 Gaze control: salience dynamics

Once the scene representation coordinates are established, it is necessary to manage the movement of the mechanical neck so that it directs the focus of attention to that position. Furthermore, given

[1] `https://www.raspberrypi.org/documentation/hardware/camera`

the existence of several faces detected and stored in the local memory of the scene, it is necessary to implement a decision mechanism that decides where to look at the next moment.

To govern the movement of the mechanical neck, salience dynamics and points of attention are introduced. These represent the faces detected in the scene. Each of them contains the position in the scene (α, β) which must be commanded to direct the neck towards it.

Salience is everything that attracts attention or that stands out in a given situation, hence the focus of attention may vary over time. In the proposed system, salience will indicate which point of attention should be visited next. Each face detected has an associated salience, which grows over time and is canceled each time it is visited. Thus, if we have a point of attention with a very high salience, it will be visited next, since it is a point that draws attention; if salience is low, it will not be visited.

One way to decide the salience of a point of attention is based on how much time has passed since it was visited. When a point is visited, its salience is set to 0. In addition, a point that has not been visited for a long time will attract more attention than one that has recently been observed. Thus, the system's behavior is similar to a human eye since, according to biology studies [W. H. Tedford et al., 1978, Jain et al., 2015], when the eye responds to a stimulus that appears in a position that has previously been observed, the reaction time is usually greater than when the stimulus appears in a new position. This effect is known as inhibition of return (IOR), which follows the equation 1.

$$Salience(t) = \begin{cases} 0 & \text{if object observed} \\ Salience(t-1) + 1 & \text{otherwise} \end{cases} \tag{1}$$

The algorithm designed allows the system to alternate the focus of the camera between the different faces in the scene according to their salience. The system considers that all faces have the same preference of attention, so they are all observed during the same time and with the same frequency. If it is necessary to assign different preferences of attention to the faces, different growth rates for salience can be set. This would cause the neck to remain longer on the faces whose salience grows faster.

The problem of how to revisit a point from the spatial interpretation was also approached, since it is a problem of evolution of the hypotheses in the period of time between detections in t and $t+n$ (where n is the time since a point has not been revisited). It is assumed that a detected face will be found again in the vicinity of where it was previously observed.

2.4 Face tracking

When the gaze distribution system chooses a face, it will look at it for 3 seconds; even following it spatially if the face moves. For this tracking, in order to avoid excessive oscillations and to have more precise control over the mechanical neck, a P-controller was implemented to control the speed in pan and continuously keep the face target in the center of the image. This P or proportional controller allows high speeds to be commanded to the neck if the focus of attention to which it must be directed is at a great distance from the current position, or low speeds if small corrections are required.

$$v(Pan) = \begin{cases} 0 & \text{if} \quad \epsilon_p < 0.3 \\ K_p \cdot (P_t - P) & \text{if} \quad 0.3 \leq \epsilon_p < M_p \\ K_p \cdot M_p & \text{if} \quad M_p < \epsilon_p \end{cases} \tag{2}$$

The controller follows Equation 2, where: K_p is the P control gain, P_t is the Pan of the target, P is the current Pan, M_p is the maximum Pan acceptable error and ϵ_p is $(P_t - P)$. The output graph of the P-controller at different positions of the point of interest is shown in Figure 6.

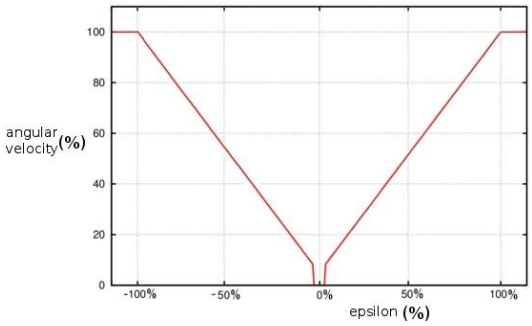


Figure 6: P-controller mechanism

2.5 Exploring new faces

At any time, and in a systematic way, the search for new faces on the scene may be of interest. To do this, scan points or virtual faces with high salience are periodically inserted (every `timeToSearchFace` time) in the local memory. This search may be of interest, above all, at the beginning of the execution, at which time the areas of the scene containing faces to follow are still unknown.

The exploration points can be of two types: random or systematic ones. The generation of the former consists of assigning completely random coordinates $(pan, 45)$, within the range of travel of the mechanical neck ($pan = [-160, +160]$, tilt is fixed at 45). The systematic exploration points will ensure that all areas of the scene will be supervised during the execution. Thus, these points will go from the lowest pan position to the highest position, and also with the fixed tilt coordinate. The random exploration points will ensure that last areas of the scene to be supervised during the systematic exploration can be randomly visited.

The points of interest, whatever their type, will have a high initial salience so they can be visited more quickly and thus it can be checked whether they contain faces. If this is the case, these virtual points or faces will continue to exist, as they will become real faces and will be treated as such. This is how new faces enter the system: they are inserted into the memory of faces and enter the dynamics of gaze distribution.

There will be a great proliferation of virtual faces at the beginning, since this is when it is most interesting to look for faces in the scene, given that the system starts from the absolute ignorance of the environment. As it discovers faces, the desire to explore new areas will decrease proportionally to the number of these, according to Equation 3.

$$timeToSearchFace = faceCounter * searchTime \tag{3}$$

2.6 Representation of the environment: life dynamics

As already mentioned in previous sections, the visual attention system will always be guided by the tracking of faces within the scene. It can track multiple people it has previously detected and stored in local memory, alternating between them, even if they are not within the immediate field of view of the camera. People move and eventually disappear from the scene, so they must be removed from the system to keep the representation of the scene consistent with reality.

To accomplish this task of forgetting old faces, the dynamic known as life was implemented. This mechanism makes it possible to know whether an object has left the scene or is still in it. Its

operation is inverse to that of salience; that is, an object frequently visited will have a longer life than one that has scarcely been visited. If the life of an object is less than a certain threshold, it will be discarded and will not be visited again.

To implement this dynamic, each time an object is visited, its life increases a little, with a maximum limit to avoid saturation. The life of unobserved objects will diminish over time. Thus, if the life of an object is longer than a certain threshold, it is still on the scene. However, if it is below the threshold, it disappears. To forget such old elements, the *life* dynamics follows Equation 4.

$$Life(t) = \begin{cases} min(MAX_{LIFE}, Life(t-1) + \Delta) \\ \qquad \text{if object observed} \\ Life(t-1) - 1 \\ \qquad \text{otherwise} \end{cases} \tag{4}$$

Where Δ is a bonus factor used when the element is a human face, because this kind of element it is considered more relevant for the visual attention system.

3 Software infrastructure

The attention mechanism described was implemented on the `PiBot` robotic platform described in [Vega and Cañas, 2018], and it uses the open vision library described in [Vega and Cañas, 2019], where the first attempt to detect a single person was also introduced. On this platform, behavior is governed by the joint action between perception and action. Both are divided into small components.

The proposed attention mechanism was included as a further perception component (`piFollowFaces`) of this architecture, which is responsible for constructing and anchoring a representation of the scene. This representation is made up of a set of human faces belonging to the robot's environment. As the camera alone does not cover all the space around the robot, not all possible faces in the scene can be detected at any given time by the camera. This perceptual component therefore entails an active perception which moves the servo-pan unit in order to search for objects and keep its internal representation constantly updated. In other words, the rule here is to act to perceive, as opposed to perceive to act.

Objects in the robot's environment guide the movements of the camera, so the attention mechanism is bottom-up. The only top-down mechanism that exists is that the relevant objects are those with the appearance of a human face. This tendency to look towards faces is similar to the predisposition in animals towards certain stimuli depending on the context, which was detected by ethologists [Newcomb et al., 2015].

The visual attention system proposed was implemented following a state machine design (Figure 7), which determines when the steps described in Section 2 are executed. Four states can be distinguished:

1. Choose next face to be followed (State 0)
2. Complete saccadic movement (State 1)
3. Analyze image to find possible faces (State 2)
4. Follow detected face (State 3)

Initially, over time, the possible faces already stored in memory are updated. Firstly, the system checks whether any face is already out of date —because its life is below a certain threshold— and secondly, salience is increased and life is decreased.

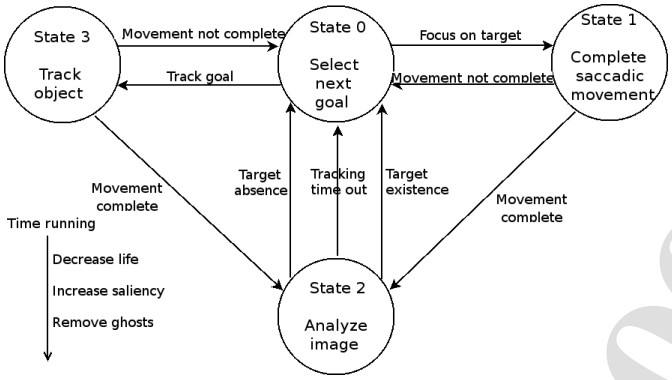


Figure 7: Finite state machine attention system

Starting from the initial state or State 0, the system asks whether there is a target to look at (in case it has a face previously stored in memory); if so, it will go to State 1; otherwise, a virtual face is created and inserted into memory, and it goes back to State 0.

In State 1, the task is to complete the movement until the absolute position is reached, which is indicated by State 0. Once there, the system will go to State 2, where the image is analyzed to detect new faces. In all cases, from here it goes to State 0 and starts over.

From State 0, it will only go to State 3 if a new face was found in the last transition, in which case it can be tracked. This is precisely the purpose of this state.

4 Hardware platform

As mentioned in Section 3, the proposed system was tested over a real robotic platform, the `PiBot` (Figure 8), which is based on a Raspberry Pi 3B+ as CPU and permits the use of a camera.

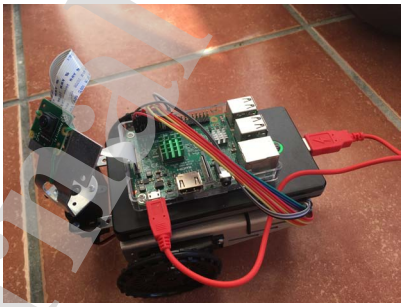


Figure 8: The robotic platform based on a Raspberry Pi board

The `PiCamera`[2] is the camera model used in this prototype, which is mounted on a pan unit. The main technical details are included in Table 1.

The pan unit is mounted over a servo model Feedback 360° from the Parallax company, which permits freedom of movement $[+180, -180]$ in pan, capable of developing a bidirectional, contin-

[2]https://www.raspberrypi.org/documentation/usage/camera/python/README.md

PiCamera parameters	Values
Sensor type	Sony IMX219PQ[7] CMOS 8-Mpx
Sensor size	3.674 x 2.760 mm (1/4" format)
Pixel Count	3280 x 2464 (active pixels)
Pixel Size	1.12 x 1.12 um
Lens	f=3.04 mm, f/2.0
Angle of View	62.2 x 48.8 degrees
SLR lens equivalent	29 mm

Table 1: PiCamera (v2.1 board) technical intrinsic parameters

uous, feedback-controllable and low-load rotation from -120 to 120 RPM. The system power is supplied by a $20,000$ mAh battery, which is also added to the chassis.

As mentioned in Section 2.5, pan moves within range of travel of the mechanical neck ($[-160, +160]$), and tilt is fixed at 45. Every time the camera is moved with respect to several axes, camera matrices must be multiplied again and again. Thus, the following steps are needed for a complete translation of the camera:

1. The camera is mounted over a Pan unit (servo), and this unit is moved along the Z axis with respect to the base of the robot (which is on the ground level).
2. Pan axis is rotated with respect to the Z axis according to the Pan angle (Equation 5).
3. The Pan support is also rotated with respect to the Y axis according to the Tilt angle (Equation 6), needed to perceive close objects.
4. Finally, the optical center of the camera is translated in X and in Z with respect to the Tilt axis.

$$\begin{bmatrix} cos(\theta) & -sin(\theta) & 0 & X \\ sin(\theta) & cos(\theta) & 0 & Y \\ 0 & 0 & 1 & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$\begin{bmatrix} cos(\theta) & 0 & -sin(\theta) & X \\ 0 & 1 & 0 & Y \\ sin(\theta) & 0 & cos(\theta) & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

5 Experiments

The main goal of these experiments was to analyze the behavior of the system according to the number of faces in the scene. In the first experiment some performance parameters are shown, in the second one, the tracking function was tested with a single face, while in the third one, the distribution of gaze (salience) and the forgetting of old faces (life) was analyzed.

5.1 Performance parameters

Firstly, two were the applications compared to capture images: *fswebcam* and *raspistill*. They were chosen because of its simplicity, being a simple command-line app for Linux-based systems.

Different resolutions were tested (1600×1200, 1280×960, 640×480, 320×240) and the results were as following: *raspistill* works faster than *fswebcam*.

Res. (px.)	***fswebcam* (ms.)**	***raspistill* (ms.)**	***videoCapture* (fps)**
1600×1200	67	58	33
1280×960	60	52	36
640×480	53	48	43
320×240	51	45	52

Table 2: Average iteration time and framerate when capturing images with different resolutions

The module in charge of capturing images runs in iterations. The resolution that works best for recognizing faces, and is not slow to process, is 640×480. The average iteration time was 48 ms. Although the highest resolution yielded better results in poor lighting conditions, this was ignored, since the natural conditions in which our system will work is with good lighting conditions. A comparison can be seen in Table 2.

Secondly, the framerate was tested. A simple Python program was implemented for this comparison. Using the OpenCV `videoCapture` function and Time package to estimate times, we found that, setting the resolution to 320×240, the PiCamera ran on average at 52 fps. But 43 fps (640×480) is a framerate enough for the system, whose performance parameters are described below.

Finally, the whole system was tested. It took 84 ms. to capture the current image (640×480), detect a face and save it on memory. However, according to the framerate results (and limited by processor computing capacity), when the algorithm orders the pan to complete a saccadic movement to obtain a snapshot of an area of the scene, the proposed system took 113 ms.

5.2 Tracking a single face

In this first experiment, the system starts, as always, with 0 faces detected. Thus, the system has to command the pan unit to perform saccades in search of human faces throughout the scene (Figure 9-a). These movements are short, precise and fast; just enough time to examine whether or not there is a face in the current image received with the camera. After a certain time, the system detects a face, the only one present in the environment.

Figure 9: Tracking a single face in the scene

With a single face (Figure 9-b), the tracking is done with smooth movements, depending on the operation of the P-controller discussed in Section 2.4. If the detected face performs sudden

movements away from the centroid of the image perceived, the movement of the pan unit is fast (Figure 9-c); when it moves slowly, the movement is more parsimonious. Thus, in Figures 9-b and 9-c, the camera appears aligned with the face it is following.

In addition, and as mentioned in previous sections, when the forced exploration time elapses ($5sec.$), the system orders the pan to complete a saccadic movement to obtain a snapshot of an area of the scene; not finding a face, it immediately goes back to the only face it has already displayed. In this case, this time is not increased since the system has only detected one face. Therefore, the described process of searching for new faces combined with tracking the detected face is repeated over time.

5.3 Tracking several faces

Figure 10: Initial scene of tracking several faces experiment

In the second experiment (Figure 10), the system also starts from having no face detected. When it detects the first face (Figure 11-a) and the forced scan time is elapsed, the system performs a forced scan through the scene. This process is repeated for a certain time, until it finds a second face (Figure 11-b). It is now when the system can continue looking at both detected faces.

Additionally, since there are already two faces in memory, the forced scan time is raised to 10 seconds. This allows the system to keep its attention on the faces detected for a longer time, as well as to continue looking for other possible ones. Thanks to this mechanism, the system is plausibly capable of finding all the existing faces in the scene, so that, after a certain time, the system finds the third and last face present in this experiment (Figure 11-c). Then, the forced scan time is raised again another 5 seconds, so it rises to 15 seconds.

The result achieved with this increase in the forced search time is that, as new faces are detected, new ones are searched for less frequency. However, when it is time to make a forced exploration, this is done, returning later to the last face it was focused on and following it even it if moves, as shown in Figure 13 according to the second scenario (Figure 12).

The behavior of the system, achieved through the two concurrent dynamics already explained above —salience and life—, is shown in Figure 14. The situation illustrated in Figure 14-a corresponds to when the system has two faces detected. In such a situation, it can be seen how the

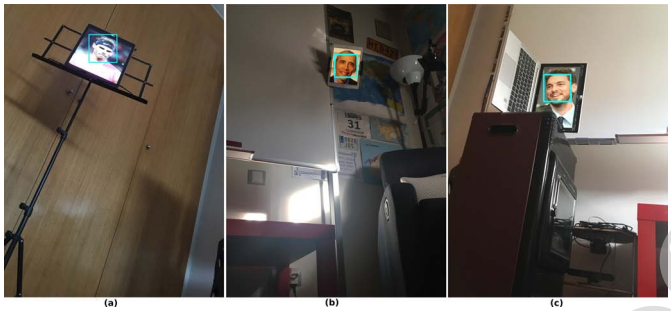


Figure 11: Tracking several faces in the scene

Figure 12: Scene of tracking several faces when they moved

salience of both faces evolves. When the system is following a face (blue) its salience decreases, while the other face stored in memory (red) increases until it wins the competition and forces the system to look at it.

The evolution of the life of both faces, when both remain in the scene, can be seen Figure 14-b. Its operation is inverse to salience; that is, each time the system visits a face, its life increases a little, with a maximum limit to avoid saturation.

Figure 14-c shows a situation in which a face is occluded in the scene (red), so the system stops detecting it and, therefore, its life begins to descend. When its value falls below a certain threshold, that face is discarded and not visited again.

6 Conclusions

This work presents a visual attention system, the purpose of which is to find and follow human faces in the scene surrounding a robot. To do this, a concurrent dynamic mechanism between life and salience was developed, in which the face with the greatest salience is the next point of interest

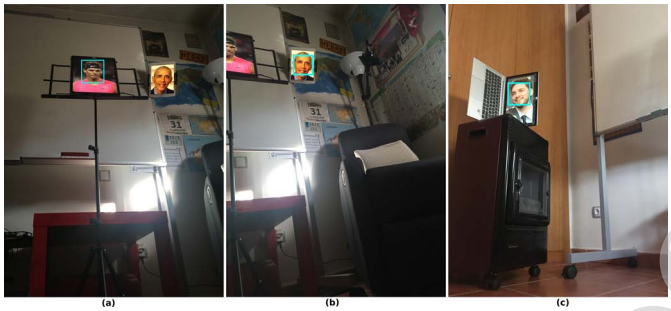


Figure 13: Tracking several faces even when they moved

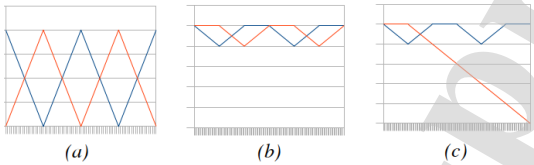


Figure 14: Salience and life evolution

to be visited and, therefore, the one that directs the servo-pan unit movement at all times. In this way, the main goal is achieved: making the robot follow all its interlocutors with its *eyes*, so that the person-robot interaction is more natural. The life dynamic enables the system to have a coherent representation of the faces in scene, thus preventing the robot from paying attention to people who are no longer there.

In addition, and since the scene is larger than the immediate field of view of the robot's camera, a local short-term memory was also implemented. This component permits the interaction between the robot and the people around it, since they may be located in positions that the robot is unable to see at a given moment but in which it knows there are human faces.

The different experiments conducted show the behavior of the global system according to these mechanisms described above, which was found to be quite similar to how a human pays attention. For example, when there is only one face in the scene, the system pays full attention to it, following it with its gaze, while looking for others from time to time. However, when there are several people, it alternates its attention between all of them and the exploration time increases proportionally. Contrastingly, when there are no faces, this time is minimal, with the behavior emulating the desire to detect faces.

It is also worth highlighting that the faces that disappear from the scene are forgotten, thus avoiding ghosts in the representative memory of the environment. However, several unsuccessful attempts have to be made before a face can be considered to have disappeared, since it may sometimes not be detected due to sporadic occlusions, although the detection algorithm presented is usually quite robust to different lighting conditions and person-robot distance.

It is expected that the system developed will contribute in the short term to develop an intelligent robot, which includes a larger computational capacity and a more complex sensory system, to solve the robot-human interaction issue at general level in a real environment. The platform could be the semi-humanoid robot Pepper, manufactured by Aldebaran Robotics, which was recently acquired from the Robotics Group at Rey Juan Carlos University.

One of the possible improvements to this work might be the recognition not only of a human face, but also of expressions and their subsequent correspondence with emotions. This process is complex, since, apart from detecting the face, it is necessary to detect the elements that influence the formation of the different facial expressions, and later identify the emotions corresponding to such expressions. New embedded acceleration boards, such as Jetson Nano, Google Coral or Intel NCS, are the perfect candidates to be incorporated into the PiBot platform and, with this, enable the execution of this type of facial and emotional recognition algorithms with advanced Deep Learning techniques in a low-cost platform like this one.

Funding

This work was partially funded by the Community of Madrid through the Jóvenes Investigadores project (ref. F664) of the Rey Juan Carlos University.

References

[Bajcsy et al., 2016] Bajcsy, R., Aloimonos, Y., and Tsotsos, J. (2016). Revisiting active perception. *Autonomous Robots*, 42.

[Bakar and Mohamad Amran, 2015] Bakar, M. and Mohamad Amran, M. F. (2015). A study on techniques of person following robot. *International Journal of Computer Applications*, 125:27–30.

[Ben Amor et al., 2014] Ben Amor, H., Neumann, G., Kamthe, S., Kroemer, O., and Peters, J. (2014). Interaction primitives for human-robot cooperation tasks.

[Biswas, 2016] Biswas, P. (2016). *Exploring the use of eye gaze controlled interfaces in automotive environments.*

[Ci and Huang, 2016] Ci, W. and Huang, Y. (2016). A robust method for ego-motion estimation in urban environment using stereo camera. *Sensors*, 16(10).

[Clark and Stark, 1975] Clark, M. and Stark, L. (1975). Time optimal behavior of human saccadic eye movement. *IEEE Transactions on Automatic Control*, 20(3):345–348.

[Ewerton et al., 2015] Ewerton, M., Neumann, G., Lioutikov, R., Ben Amor, H., Peters, J., and Maeda, G. (2015). Learning multiple collaborative tasks with a mixture of interaction primitives. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1535–1542.

[Faraj et al., 2020] Faraj, Z., Selamet, M., Morales, C., Torres, P., Hossain, M., and Lipson, H. (2020). Facially expressive humanoid robotic face. *HardwareX*, page e00117.

[Jain et al., 2015] Jain, A., Bansal, R., and Kumar, A. (2015). A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students. *International Journal of Applied and Basic Medical Research*, 5:122.

[Li et al., 2016] Li, X., Xu, Q., Li, B., and Song, X. (2016). A highly reliable and cost-efficient multi-sensor system for land vehicle positioning. *Sensors*, 16(6).

[Luong et al., 2015] Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

[Menéndez et al., 2013] Menéndez, B., Cañas, J., Perdices, E., and Vega, J. (2013). Programming a humanoid social robot using the jderobot framework. In *RoboCity2030 11th Workshop, Robots sociales, pp 71-94, U.Carlos III, March 14th 2013. ISBN:978-84-695-7212-2.*

[Newcomb et al., 2015] Newcomb, T., Turner, R., and Converse, P. (2015). Social psychology: The study of human interaction. *Social Psychology: The Study of Human Interaction*, 19:1–591.

[Nobre, 2015] Nobre, K. (2015). *The Oxford handbook of Attention.*

[Parks et al., 2014] Parks, D., Borji, A., and Itti, L. (2014). Augmented saliency model using automatic 3d head pose detection and learned gaze following in natural scenes. *Journal of Vision Research.*

[Rautaray and Agrawal, 2015] Rautaray, S. S. and Agrawal, A. (2015). Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review*, 43(1):1–54.

[Robinson, 1964] Robinson, D. A. (1964). The mechanics of human saccadic eye movement. *The Journal of Physiology*, 174(2):245–264.

[Vega and Cañas, 2018] Vega, J. and Cañas, J. (2018). PiBot: An open low-cost robotic platform with camera for STEM education. *Electronics*, 7:430–446.

[Vega and Cañas, 2019] Vega, J. and Cañas, J. (2019). Open vision system for low-cost Robotics education. *Electronics*, 8:1295–1315.

[W. H. Tedford et al., 1978] W. H. Tedford, J., Hill, W. R., and Hensley, L. (1978). Human eye color and reaction time. *Perceptual and Motor Skills*, 47(2):503–506. PMID: 724388.

Author biography

Associate Professor of Telematic Systems and Computing at Universidad Rey Juan Carlos (URJC). PhD (hon., int.) in Computer Science and A.I. by Universidad de Alicante (research also done at URJC), MEd in Teaching by URJC, BASc in Computer Science by Universidad de Extremadura and MSc in Computer Science by URJC, with studies also taken at Universidad Politécnica de Madrid.

My research covers a wide range of topics about Robotics. A part of it has been carried out at York University of Canada and in different colleges of the University of Eastern Finland. At the beginning of my doctoral thesis research, I was working on a robust autonomous robot —previously developed as a guide-robot— which used a visual memory to navigate. These algorithms were finally integrated into a new educational framework and adapted to run in a robotic platform, called PiBot, which was completely developed from scratch.

Conflict of Interest

The author declares no conflict of interest.