People management framework using a 2D camera for human-robot social interactions

Jacques Saraydaryan[1,2], Raphael Leber[1], and Fabrice Jumel[1,2]

[1]CPE Lyon, [2] CITI Lab., INRIA Chroma, Université de Lyon, Villeurbanne, France

Abstract. In order to perform tasks and offer socially acceptable human-robot interactions, domestic robots need the ability to collect various information about people. In this paper, we propose a framework that allows the extraction of high-level person features from a 2D camera in addition to tracking people over time. The proposed people management framework aggregates body and person features including an original pose estimation using only a 2D camera. At this time, people pose and posture, clothing colors, face recognition are combined with tracking and re-identification abilities. This framework has been successfully used by the LyonTech team in the RoboCup@Home 2018 competition with a Pepper robot from SoftBank Robotics where its utility for domestic robot applications was demonstrated.

1 Introduction

Interacting with humans in populated environments is a very challenging task for social robots. Many tasks, like people identification, intention recognition, navigation in crowed environments or situation description need to be performed by domestic robots.

The RoboCup@Home (part of the RoboCup international robotic competition) intends to evaluate current domestic robots through real life scenarios. The focus lies on human-robot-interaction, navigation and mapping in dynamic environments, computer vision, object manipulation and adaptive behaviors. A set of benchmark tasks is used to evaluate the robots' abilities and performance in realistic home environment settings. For example, during the "Party Host scenario" trials, robots provide general assistance to guests during a party (welcome, introduce a new guest to others, describe guests to the bartender, escort an exiting guest to a cab ...).

Such interactions require abilities such as following an operator, finding a specific person in a crowd or comprehensibly describing a person. High level information about people (detected people, pose estimation, body description, clothing description...) has to be extracted from raw data provided by robot sensors.

As some characteristics (typically a person's positions) vary over time, the use of the tracking approach, more precisely, Multiple Object Tracking (MOT) [1], is needed. In recent works, the Deep Learning approach applied on large dataset has allowed the extraction of numerous human features (e.g. clothes color, gender, hat, hair color) [2, 3]. All these approaches give important clues and tools to characterize and track people and could be based on simple sensors such as a 2D camera.

In the case of a domestic robot, we need a framework able to provide all these features with only onboard sensors. A modern approach would be to define all the characteristics needed and train a neural network. Unfortunately, at this time, the creation and labeling of such a large and complex dataset is not possible. The only practical approach is to aggregate different features extraction tools (mostly based on deeplearning) and merge them. For example, a RoboCup@Home team developed a general tracking tool for MOT called "wire" [4]. Another team defined a specific framework for "Person-Following" tasks [5] based on OpenPose tools [2] and color features extraction.

Relevant works have been made on MOT applied to people tracking [1], but few of them are from a human (or robot) eye's perspective (e.g MOT16). Most MOT proposals consist of tracking scenes through a surveillance camera that is positioned much higher than the human eye. When people disappear and reappear, trackers need to re-identify people and associate them with a previous identity. This process, called Person Re-Identification (PReID) [6], uses different collected persons characteristics. In the context of domestic robots, the problem of re-identification is important, but in order to achieve various tasks, abilities to extract goal oriented people characteristics is essential. In this paper we propose a framework allowing the extraction of people's high-level information from a 2D camera and the tracking of people over time for future interactions. The proposed people management framework (available on github[1]) aggregates body and person features. At this time, people pose and posture, clothing colors, face recognition are combined with tracking and re-identification abilities. The paper is composed as follows: the section 1 talked about the needs of people management abilities for domestic robots. The section 2 presents an overview of our people features extraction and tracking framework. The people features extraction processes are detailed in section 3. A proposed people tracking system is described in section 4. The section 5 presents results on pepper robot experiments. Finally, future works and improvement are discussed in section 6.

2 Overview of the People Management Architecture

As presented in section 1, managing social scenarios, with direct and indirect people interactions, involves to characterize, track people and understand their intentions.

To do so, we provide a framework collecting people information and extracting high-level semantic data. These semantic data, goal oriented, are crucial to achieve social robots scenarios.

The figure 1 presents the different framework blocks. First of all, the robot gets scene information through camera (2D picture). This image is processed in

[1] https://github.com/Robocup-Lyontech/People-Management-Framework

order to extract each person in the scene (OpenPose [2]). Semantic information are added concerning people posture, hand posture and estimated position (Pose Extraction). Then information such as face and clothes are used to identify a person (face detection, color detection and naming). Finally, all semantic information are gathered and sent to the robot decision process. Robot can identify people, understand intentions (e.g hand call) and has an overview of the scene.

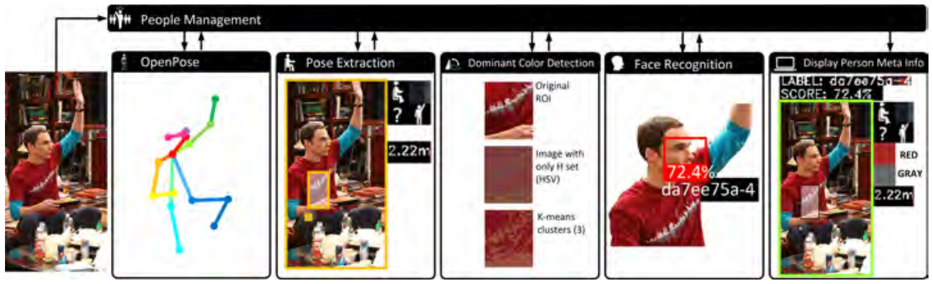

Fig. 1: People Management Architecture

Let a detected person noted $p \in P$, the following features are associated such as $p = (p^{face}, p^{shirt}, p^{trouser}, p^{posture}, p^{hand\_posture}, p^{pose})$ where :

- $p^{face} = (label, score)$ where $label$ and $score \in [0, 1]$ representing respectively the id of the detected face and the probability of matching with previously learned faces
- p^{shirt}, $p^{trouser}$ define the dominant colors of portion of the shirt and portion of the trouser. Each of theses items is a list of objects composed of a color name, the RGB value, the HSV value and the percentage of the portion of the targeted object (shirt or trouser). e.g $p^{shirt} = \{p^{metaC_0}, p^{metaC_1}, ..., p^{metaC_m}\}$ where $p^{metaC_i} = (p^{rgb_i}, p^{hsv_i}, p^{name_i}, p^{score_i}), \forall i \in [0, m]$
- $p^{posture} \in \{standing, sitting, lying, undefined\}$ is the label of the detected posture
- $p^{hand\_posture}$ is an array of 2 objects (Left and Right arm), each object represents the posture of the arms $p^{hand\_posture} = (p^{hand_L}, p^{hand_R})$ such as $p^{hand_i} \in \{point\,left, point\,right, call, crossed, undefined\} | \forall i \in \{L, R\}$
- $p^{pose} = (x, y, \theta)$ is the estimated pose (position and orientation), expressed in a "top-view" map with the robot as the origin.

All those information are managed by the **People Management** block in charge of scheduling and launching people features detection blocks.

Indeed, the following workflow is executed: As soon as the **Openpose** block detects people, the **Pose Extraction** block is triggered. Information such as people pose (standing, sitting, ...), people bounding box, shirt and trouser regions of interest (ROI) are then available. These ROI are used to extract main colors of shirt and trouser through the **Color Detection** block. The people bounding box becomes the input of the **Face Recognition** which identify people if they were already seen, register the current face otherwise. At the end, the **People Management** block gathers all these features to each detected people and publish information.

People and associated features are then collected by an additional block: the **People Tracker**, that creates and maintains a set of tracked persons over the time.

When information of observed person p at a given time t is collected, a score is computed for each p regarding to a list of already tracked people T_i. This score combines person attributes e.g face, color and pose score. If the score is upper a threshold the current observed person p update the tracked person T_i, otherwise a new tracked person is created. A forget function removes tracked persons if they are not updated with new observations.

Sections below detail each processing block of the framework.

3 People Features extraction

3.1 People pose and posture

For each detected person, OpenPose outputs a list of body parts, each given with a confidence and a point on the image. Those points are expressed in 2 dimensions but the third dimension is crucial to get people pose on a map, and gives more information for people posture. To compensate the lack of a depth measurement, we will explain how we lay on a main hypothesis and on a calibration dataset. The hypothesis is that the camera horizontal field of view is parallel to the ground. This hypothesis will be used in the computation of p^{pose} and $p^{posture}$. The calibration dataset will be used to estimate people distance with linear interpolation. Therefore the calibration enters in the computation of p^{pose}. The calibration dataset is made as follow : we recorded at every meter, an average size person, straight on his legs and facing the camera, in order to maximize the limbs components on the 2D camera plane.

People pose p^{pose} is computed in four steps. The first step is to estimate the depth of the persons ($p^{pose,x}$) in the image. To do so, we consider that at least one limb (e.g : one front leg, one arm, one flank, ...) is seen with most of its components on the image plane (depth component small enough to be neglected) We find such limb $limb_{ref}$ in order to perform a distance evaluation. To find it, we normalized each limb size with its respective calibration limb size. Let call $limbs_{normalized}$ the list of normalized limbs. As we miss one dimension (image depth) we know that the biggest normalized limb is the one with the most components in the image plane, as it is in calibration data. Then we do $limb_{ref} = \max(limbs_{normalized})$ to select the biggest normalized limb as a reference to evaluate distance.

Once $limb_{ref}$ is found, it is compared to calibration data in order to find with the closest two calibrated distances. A linear interpolation is done between the two closest calibration data, in order to find $limb_{ref}$ distance ($d_{limb_{ref}}$), and therefore the measured body distance.

This distance is set in $p^{pose,x} = d_{limb_{ref}}$. Then we approximate $p^{pose,y}$ with equation 1

$$p^{pose,y} \sim p^{pose,x} * sin(\frac{H_{FOV} * (p^{neck,x} - (i_w/2))}{i_w}) \tag{1}$$

with H_{FOV} the camera horizontal field of view, i_w the image width and $p^{neck,x}$ the horizontal coordinate of a person's neck on the image.

To complete pose on the map, equation 2 computes their orientation based on right and left confidence of people body parts (face and shoulder only). People front or back side are defined by shoulder sides and/or nose presence. Depending on front/back side, we compute α ($-\pi/2$ or $\pi/2$) and β (0 or π) in order to get an orientation angle $p^{pose,\theta} = \alpha * \psi + \beta$

$$\psi = \frac{\sum bodypart\_confidence_{right} - \sum bodypart\_confidence_{left}}{\sum bodypart\_confidence_{right} + \sum bodypart\_confidence_{left}} \tag{2}$$

Posture :

With a limited 2D information, multiple combinations could match with one of each posture. As well, a same combination could match for all the postures. To solve the first problem we use a scoring system on different combinations To fix the second problem we considered the hypothesis mentioned for pose detection. Here are the evaluation criteria for each position, starting with the body. "Standing" relies on xy thigh ratio and neck position on image height (i.e. hypothesis). "Sitting" checks height ratio of thigh and calf as well as knee angle. "Lying" test looks at legs xy ratio and also at head (each point) position on image height. About hands, "Pointing left/right" and arm "Crossed" compute forearms and clavicles as vectors. Hand "Call" checks wrist above shoulder or elbow above nose.

3.2 Color Detection

We optimized our previous color detection system [7] to give additional information to the **People Management** block. Although our color detection is still based on kmean-clustering on the H of the HSV image color, our system extracts dominant colors and associated readable names (E.g Red, Dark Blue,...). To do so, main HSV color value is converted in RGB value given fixed S and V values (only the Hue value is considered). Then, the closest X11 color is used [8] to associate a name.

Information about saturation and value intensity is relevant to express darkness and extreme value such white and black. Based on the HSV color representations, thresholds on saturation and values are used to complete color name.

Future integration of deep learning color naming based approaches [9] will help to improved common issues concerning illumination and textures.

3.3 Face Recognition

The face recognition mechanism is based on the method provided by Adam Geitgey's based on the ResNet-34 of [10] library[2]. By adding automatic face learning if no matching exists, our method allows a dynamic people face learning and recognition. Using a set of face recognition options (HOG [11], Haar Cascades [12], bounding box from OpenPose), the learning face phase can be adapted according to the need. As a result, when the service is targeted with an image, the system return per detected face a percentage of matching of the more relevant learner face or unknown otherwise. If no known face is detected, the current detected face is learned automatically.

4 People Tracking

All persons with associated features need to be followed and tracked over time. Thus, a robot could keep a state of met people, called tracked people, that it could use for future tasks. The **People Tracker** block is a Multiple Object Tracker (MOT), that need, according to the literature [1], to cover 2 objectives: What is the number of objects ? How to maintain object identities ?

In the different targeted scenarios, the number of people is not previously known, tracked people is maintained during a time until to be removed by the system in accordance with the forgetting function. Identities of tracked people are fed by people observations that update the tracked people states. The tracked people update is triggered using a similarity function, resulting of the summation of multiple cues, between people observations and tracked people set. The greedy approach is used to associate observation with the most similar tracked person.

This state of met people is defined by a tracked person, noted $T_i \in T$ where T is the set of tracked persons. A tracked person T_i is composed of a set of person features like face, color and pose information updated with people, p, attributes when associated to T_i. Additional information such as a *weight* (how many time a detected person p is associated with the tracked person T_i), *lastUpdateTime* (referring to the last time a detected person p is associated with the tracked person T_i) are used to keep or forget a tracked person

When the tracker receives detected people p, it computes a score of similarity between p and T_j based on attributes, e.g $face$, *shirt_color*, *trouser_color* and *pose*. The similarity score (called $general_{score}$) is defined as follows for the given attribute set:

$$general_{score}(p,T_j) = w_0.face_{score}(p,T_j)+w_1.color_{score}(p,T_j)+w_2.pose_{score}(p,T_j) \quad (3)$$

The different $general_{score}$ weights w_k can be adjusted according to the needs: if the pose information rate is high the importance of w_2 will increase. If the target scenario needs to identify people with specific color, the w_1 score weight

[2] https://github.com/ageitgey/face_recognition

should be higher. Furthermore, score weights could be computed using a Boosting based algorithm [1](e.g AdaBoost) given a training labeled dataset.

If the similarity score is over a threshold, the person p updates current tracked person T_i such as $T_i = \underset{T}{argmax}(general_{score}(p, T))$, otherwise a new tracked person is created and updated with p information. Other affections are possible using Hungarian method as used in [13]. For the rest of the article, only face, color and pose person attributes will be considered.

A tracked person T_i can be defined as follows:
$T_i = (T_i^{pose}, T_i^F, T_i^C, T_i^{weight}, T_i^{last})$ where :

- T_i^{pose} is the current tracked person pose,
- T_i^F contains information about the face of the tracked person, $T_i^F = \{T_i^{metaF_0}, T_i^{metaF_i}, ..., T_i^{metaF_o}\} \ / \ T_i^{metaF_j} = (T_i^{label_j}, T_i^{fweight_j}, T_i^{last_j})$ $\forall j \in [0, o]$ where $T_i^{fweight_j}$ represents the number of observation matching the current face and $T_i^{last_j}$ the last update date.
- T_i^C contains clothes colors information, $T_i^C = \{T_i^{hsv_1}, T_i^{hsv_2}, ..., T_i^{hsv_p}\}$

4.1 Updating tracked people

When a person p_i is associated to a tracked person T_i, the T_i attributes are updated. First of all, current tracked person weight and last update time are updated ($T_i^{weight} += 1$, $T_i^{last} = now$). This information is used to keep or remove a tracked person. T_i pose is adjusted to the observation pose values, $T_i^{pose} = p_i^{pose}$. The T_i color update process, given a queue size, adds the new main hsv color $p_i^{hsv_k}$ and removed the oldest hsv color if the queue size is reached (FIFO). Concerning face information, if the face label of the person is equal to one of T_i^F label, the weight and the last detected time of this one are updated. Otherwise, a new vector $(T_i^{label_{o+1}}, T_i^{fweight_{o+1}}, T_i^{last_{o+1}})$ is added to T_i^F such as $T_i^{label_{o+1}} = p_i^{face}$, $T_i^{fweight_{o+1}} = 1$ and $T_i^{last_{o+1}} = now$.

4.2 Forgetting outdated tracked people

In order to keep the system stable, outdated tracked people are removed. The forgetting function has to keep tracked persons for a given time from their respective updates. Moreover, tracked persons with a greater weight has to be kept longer. Considering these constraints, the following function based on classical forgetting curve is used as $forget$ function:

$$forget(T_i) = 1 + log\big(Th_t - \frac{Th_w}{min(Th_w, T_i^{weight})} * (now - T_i^{last})\big) \quad (4)$$

where Th_t define the maximum time that a tracked person would be kept if no update occurs, and Th_w is a weight threshold over which the tracked person would be kept during the maximum time. Moreover Th_w and Th_t can be customized for VIP tracked persons (for example operator in a people following task).

A tracked person updated regularly in the past will be kept by the tracker from its last update until the Th_t value (only if $T_i^{weight} \geq Th_w$).

Finally, given T_i, for a forget function value $forget(T_i) < 0$, the tracked person T_i is removed.

4.3 Face Similarity score

The **Face detection** block associates to each person some face labels with confidence scores. When a person face is unknown, a new face is registered. So a same person can be associated to a set of faces (e.g due to different camera/person settings). The face similarity score maintains for each tracked person T_i a set of face information T_i^F as defined in section 4.

According to the detected person p with a given face label p^{face}. The face score (noted $face_{score}$) given by the current observation, is equal to the face weight $T_i^{fweight_k}$ of the person tracked, normalized with the sum of all people tracked face weight such as $T_i^{label_k} = p^{face}$.

This function gives values for an observed face, proportional to the past associated faces weights.

4.4 Color Similarity score

When the face becomes unavailable, color information about clothes gives important clues to associate an observed person to a tracked person. Given an observed person p and a tracked person T_i, the used color score is the distance $d()$ between, for example, p^{shirt} and the average hsv color of T_i^C.

$d()$ function can be e.g. Hue distance value, Euclidean distance, one of the CIELAB $\Delta E*$ distances [14, 15]. Other image distance/similarity can be computed like Color Histogram intersection [16].

4.5 Pose Similarity score

Standard tracking system, such as Kalman filter or particle filter, are designed to track a single person according to move estimation and observed data. Knowing the pose of a tracked person, Kalman filter could be used to measure the similarity of the given measure p^{pose} and tracked person. To do so, the Kalman filter is applied on the current tracked person T_i with the observed person pose p^{pose} such as

$(T_i^{pose}{}_t, \sum_t) = KF(T_i^{pose}{}_{t-1}, \sum_{t-1}, u_t, p_t^{pose})$

where $\sum$ is the covariance error on the state and u the command applied on the system (in our case, $u = 0$ because no prior information about person's objectives is considered).

Then we define, the pose score as a distance between observation and the new state of the system. This pose score ($pose_{score}$) is also normalized in $[0, 1]$ regarding to the distances of other tracked person pose.

The Kalman filter estimation is kept only on the T_i on which the observed people p is associated. For the other tracked persons the previous state is restored.

If the system guarantees a sufficient information rate (regarding to the human velocity max e.g $5ms^{-1}$), the Kalman filter can be applied. If the data rate is too low, the motion model of the Kalman filter becomes inefficient leading a great filter drift. We propose, in the case, to use only the last person tracked position and compute a function as an incertitude propagation model of the last position.

$$fn(T_i^{pose}, p^{pose}) = \begin{cases} 1 \text{ if } d(p^{pose}, T_i^{pose}) < d(T_i^{pose}, (T_i^{pose} \pm v_{max} \times \Delta T)) \\ 0 \text{ otherwise.} \end{cases} \quad (5)$$

where $d()$ is the distance function (in our case euclidean distance), v_{max} is the maximum human velocity considered and ΔT the time elapsed since the last person tracked update, $\Delta T = now - T_i^{last}$.

The pose score function to apply on low data rate is also normalized between $[0, 1]$ regarding to other tracked person distances.

5 Experimental Evaluation

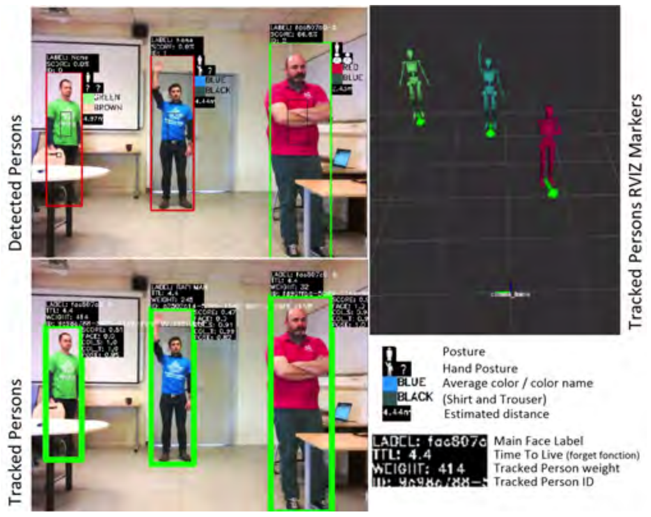

Fig. 2: Experimentation feedback

The Pepper robot from SoftBank Robotics is our evaluation platform collecting images of the scene. The camera of the pepper produces VGA (640x480) images at a rate fixed of 10fps.

Several scenarios have been tested, the presented one (a video illustrating the results here[3]) is composed of a robot at a fixed position, (with head camera

[3] https://youtu.be/0qSulBGBarg

parallel to the ground) and 3 people appearing at different time. An example of robot feedback is displayed in in figure 2.

In the first part of the scenario, a person (called person 1) is sitting in front of the robot (red shirt), after calling the robot (hand call), the person stands up and travels in the robot field of vision. Then, the person 1 sits again. In the second part of the scenario, a second person (called person 2) then appears (green shirt) and travels around the person 1. In a third scenario part, both person 1 and person 2 stand up and cross each other. Finally, a third person appears, walks and stands between person 1 and person 2. Then all people, walk in the robot direction and stand close to the robot.

During all the experiments different person features including shirt color, trouser color, person posture, hands posture are correctly identified. Some color names has some variation (especially between cyan and blue color names, same as red and pink). Few color name errors occurred. They are mainly due to a bad trouser ROI or due to the presence of the person's arm in the shirt ROI. Concerning person posture and arm posture, no errors has been identify during the experiments

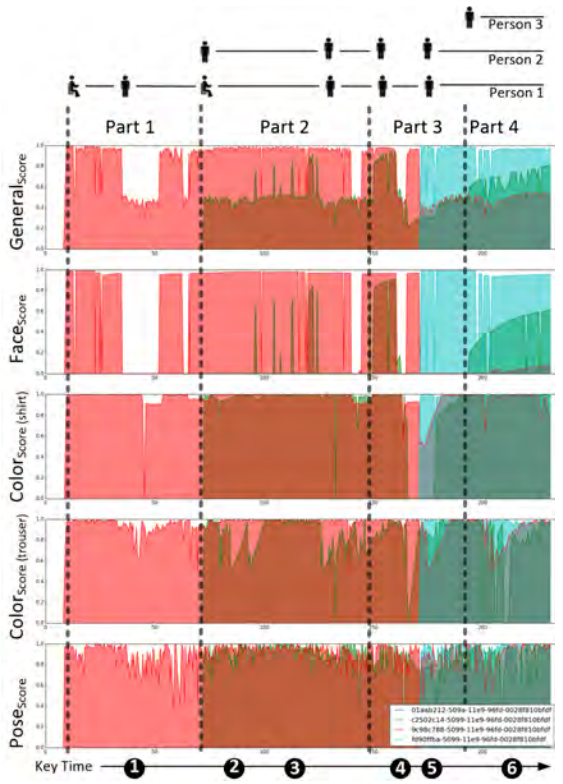

Fig. 3: Tracked people score during the experimentation

The figure 3 shows the different people tracked score according to the time.

On the top figure, the scenario situation is described. Indications such as how many people is there in the scene and what are the current people postures are available. Below all scores of generated tracked people are displayed during the time. Thus $general_{score}$ [4], $face_{score}$, $color_{score}$ and $pose_{score}$ of tracked people are available. Tracked people are identified by a dedicated color and an universally unique identifier (UUID). Finally at the bottom of the picture, relevant scenario moments are indicated.

During the part 1 of the scenario the sitting person 1 is well identify by the face detector. Moreover color and position values have no variations during the time leading to high color and pose score. As soon as person 1 turns his back to the robot, the $general_{score}$ dropped due to the absence of face identification. One can see at the key time one, that the shirt $color_{score}$ dropped down, this is due to the fact that no shirt bounding box is available.

A new person appears at part 2, the key time 2 shows that the $color_{score}$ of the trouser of the person 2 dropped and rise up two times. This variation results of bad color detection. Indeed, the trouser bounding box at the both moments is apply on chair and not on people.

At key time 3, tracked person associated with person 2 has medium face score values, increasing gradually over time. In fact, first face recognition on person 2 is bad, next one are better but due to the discontinued face recognition, the $face_{score}$ increase slowly.

In part 3, person 1 and 2 cross each other and keep their associated tracked people. At key time 4, a person leaves the scene and when he comes back at key time 5, a bad face recognition prevents the good association . Person 1 for for its part, is associated with a new tracked person. At the end on key time 6, a new person enters into the scene and is also associated with a tracked person until the end of the experiment. The present scenario shows that both people features extraction and people tracking has good results.

Finally, a primary version of our features extraction system has been tested during the RoboCup@Home 2018 competition, leading the LyonTech team to gain points to the Speech and Person Recognition scenario. The figure 4 displayed results obtaining during the competition. Despite a few bad color names processing most of information describes correctly the scene.

[4] Weights of the $general_{score}$ are chosen according the ground truth on several scenarios and results of a weight $w_0 = 0.5$ for the $face_{score}$, $w_1 = 0.15$ for the $color_{score}$ and $w_2 = 0.35$ for the $pose_{score}$.

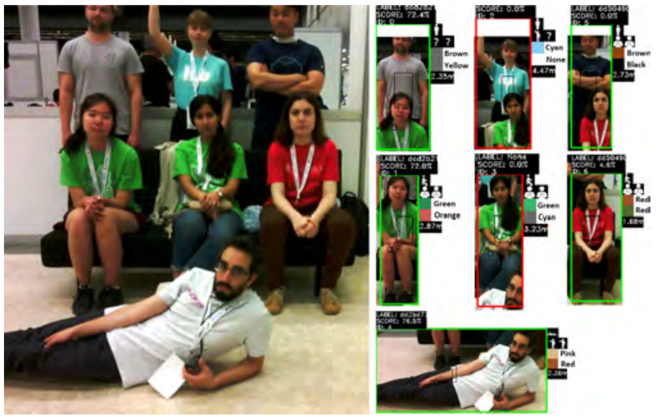

Fig. 4: Result of Speech and People Recognition scenario during the RoboCup@Home 2018 challenge

6 Conclusion

To consider robot deployment in homes, people identification, intention recognition or situation description capabilities have to be developed with cheap and general purpose sensors such as a classical 2D camera.

Our main contribution is proposing a framework which equips robots with high-level information about individuals that in turn yields more productive social interactions. The framework is driven by domestic use cases (e.g. following people, locating people, checking if people sat, looking at people pointing, describing one person among others, coming to a person calling). It manages different blocks to extract people distinctive high-level features such as clothing colors, face recognition, position and posture.

Different real-life experimentation demonstrate the usability and relevance of the framework, especially during the RoboCup@Home 2018 competition.

Global performance could be enhanced with higher resolution images (better face recognition and better position evaluation) as well as more computation power (more FPS leading to better tracking).

Some block's improvements are still in development. Regarding to the tracker, weights of the scoring system could be optimized with reinforcement or adaptive learning. Concerning the pose extraction, The dependence between people position and people height can be reduced through other features (e.g. age, gender).To go further, the framework allows an easy addition of new blocks, even for new features, in order to answer new use-cases, or to have better performance on already covered use-cases. A coming block, is the implementation of the work of our colleagues [3] to get 45 more features (e.g. Causal/Formal upper/lower clothes, carrying plastic bag, gender) of people.

References

1. Wenhan Luo, Xiaowei Zhao, and Tae-Kyun Kim. Multiple object tracking: A review. *CoRR*, abs/1409.7618, 2014, last 2017.
2. Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *CoRR*, abs/1812.08008, 2018.
3. Yiqiang Chen, Stefan Duffner, Andrei STOIAN, Jean-Yves Dufour, and Atilla Baskurt. Pedestrian attribute recognition with part-based CNN and combined feature representations. In *VISAPP2018*, Funchal, Portugal, January 2018.
4. J. Elfring, S. Van Den Dries, M. J. G. Van De Molengraft, and M. Steinbuch. Semantic world modeling using probabilistic multiple hypothesis anchoring. *Robot. Auton. Syst.*, 61(2):95–105, February 2013.
5. Minkyu Kim, Miguel Arduengo, Nick Walker, Yuqian Jiang, Justin W. Hart, Peter Stone, and Luis Sentis. An architecture for person-following using active target search. *CoRR*, abs/1809.08793, 2018.
6. Bahram Lavi, Mehdi Fatan Serj, and Ihsan Ullah. Survey on deep learning techniques for person re-identification task. *CoRR*, abs/1807.05284, 2018.
7. Fabrice Jumel, Jacques Saraydaryan, Raphael Leber, Laëtitia Matignon, Eric Lombardi, Christian Wolf, and Olivier Simonin. Context Aware Robot Architecture, Application to the RoboCup@Home Challenge. In *RoboCup symposium*, pages 1–12, Montreal, Canada, June 2018.
8. Brad Pettit (Microsoft Corporation) Steven Pemberton (CWI). Css color module level 3. W3c recommendation, W3C, 2018. https://www.w3.org/TR/css-color-3.
9. Lu Yu, Yongmei Cheng, and Joost van de Weijer. Weakly supervised domain-specific color naming based on attention. *CoRR*, abs/1805.04385, 2018.
10. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.
11. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, June 2005.
12. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, Dec 2001.
13. Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016.
14. R. McDonald and K J Smith. Cie94-a new colour-difference formula*. *Journal of the Society of Dyers and Colourists*, 111(12):376–379.
15. M. R. Luo, G. Cui, and B. Rigg. The development of the cie 2000 colour-difference formula: Ciede2000. *Color Research & Application*, 26(5):340–350.
16. Michael J. Swain and Dana H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:11–32, 1991.