2024 IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS) | 979-8-3503-6623-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/EAIS58494.2024.10569996

A Robot-Assisted Missing Child Search Approach

Abdel Rahman Alqaddoumi
School of Computing
Montclair State University
Montclair, NJ USA
alquaddoumia1@montclair.edu

Anisha Mulinti
Montville Township High School
Montclair, NJ USA
anisha.mulinti@gmail.com

Srivalli Sreya Penagamuri
School of Computing
Montclair State University,
Montclair, NJ USA
penagamuris1@montclair.edu

Yousaf Ghaly
School of Computing
Montclair State University,
Montclair, NJ USA
ghalyy1@montclair.edu

Sanjana Chitnis
School of Computing
Montclair State University
Montclair, NJ USA
chitniss1@montclair.edu

Weitian Wang
School of Computing
Montclair State University
Montclair, NJ USA
wangw@montclair.edu

Michelle Zhu* Corresponding author
School of Computing
Montclair State University,
Montclair, NJ USA
zhumi@montclair.edu

***Abstract*— In recent years, the development of intelligent robotic systems capable of detecting and tracking individuals using computer vision and artificial intelligence has gathered attention in the field of robot-assisted applications. Such intelligent and collaborative robots that operate in the vicinity of humans make significant strides in various public and private sectors. This work-in-progress paper discusses a preliminary integrated robot system that utilizes a lightweight object detection and facial recognition techniques, namely OpenCV Haar Cascade, VGGFace and other navigation and tracking packages to enable a robot to search, identify and follow a person based on real-time streaming video and sonar sensor inputs. This study establishes a preliminary model to enable seamless robot and human cooperation in unknown environments.**

Keywords—Robotics, Human-robot interaction, Face Recognition, Computer vision

I. Introduction

Our world is extremely concerned about the startling increase in the number of missing children. It is estimated that about 2,300 children go missing every day in the United States [1]. In most cases, children can be found within hours. Some are missing permanently. The first couple of hours following a child's disappearance are very important. Traditional search and rescue efforts are extremely difficult [2]. Autonomous robots have potential to improve search efficiency by utilizing artificial intelligence techniques, sensors and camera inputs. The motivation behind the use of robots is their intelligent features to assist and expedite human's rescue efforts even in unknown environments during the critical initial missing hours. In addition, robots can be deployed as a team to roam a wide unknown area collaboratively and tirelessly. Furthermore, robots are resilient to inclement weather conditions, and can reach hazardous areas that are unsafe for humans.

An overview of the state-of-the-art work for missing children search is presented first. Many technical challenges of real-time robot systems, including roaming, map building, obstacle avoidance, perception, and decision-making are also investigated. Additionally, the computing complexities arising from the large number of frames a robot has to process each second is also considered. Although our robot is equipped with the NVidia Graphic Processing Unit (GPU), some frame-reducing techniques can drop some unnecessary images to save computing time. By utilizing the Robot Operating System (ROS), some image processing and machine learning packages, we design and test an initial version of a trained mobile robot system to search, identify and track a missing child. Experiments are still ongoing to tune and optimize the performance of this robot system. In the future, we plan to leverage the system by deploying a team of robots who can divide the search areas and communicate with each other for faster and more efficient search.

II. Related Work

In recent years, a multitude of systems have been developed to address the critical issue of tracking missing children. For example, a missing child identification system, a pioneering solution amalgamating the capabilities of deep learning and multiclass Support Vector Machines (SVM) was introduced. Convolutional network was implemented as a high-level feature extractor and child recognition was done by the SVM classifier [3]. Another study was conducted on the design, implementation, and evaluation of Children Activity Tracking System (CATS). Operating under the Internet of Things (IoT) paradigm, CATS was crafted to monitor and analyze children's activities in real time. The system offered invaluable insights for parents, caregivers, and educators, marking a significant stride in leveraging technology for enhanced child safety and well-being. However, tracking children's activity through the Global System for Mobile Communications (GSM) and Global Positioning System (GPS) in real time raised privacy concerns. The sensitive data of the child should be secure and consent of the parents should be made first [4]. A dedicated IoT-based localization module was proposed to achieve a tradeoff between accuracy and privacy for missing child searches. This innovative module was designed to elevate safety measures and location accuracy in dynamic and diverse environments. However, only particular classes of the trained model can be successfully identified. Data augmentation techniques and fine-tuning were proposed to improve the performance of the algorithm [5].

Regarding image processing and machine learning techniques for missing people, the YOLOv4 (You Only Look Once), a real time object detection algorithm was used to reduce frames-per-second processing rate. Remotely Piloted Aircraft System (RPAS) technology was used to collect aerial video feed in an unmanned aircraft. The YOLOv4 algorithm created a box and text to detect the humans. This solution was invoked to identify human objects before any further processing. However, this solution did not specifically recognize a human's face [6]. Another study [6] compared OpenCV and Matlab for face recognition and the results concluded that OpenCV was much faster than Matlab. OpenCV Haar Cascade Classifier and TensorFlow were utilized with a long training process. The results also suggested that the accuracy of face recognition using static images was higher than using a real-time video feed. However, we have to deal with real-time video in reality [7]. Face mesh and face landmarks were aspects used in a study for face detection. Face landmark detection was a computer vision task where the key points in the human face were detected. The research used a media pipe to detect the face key points and constructed the face mesh for feature maps. The model identified the face by comparing the facial landmarks of the marked face with those stored in the training database. However, the model was unable to recognize the face because of the database's non-availability of matching face landmarks [8]. The failure to recognize the face occurred due to the absence of matching facial landmarks in the training database. The lack of available data for the specific facial features led to the unsuccessful identification process.

A study on face detection used the WIDER FACE dataset, which is currently the largest face detection dataset [9]. The WIDER FACE dataset is a subset of the WIDER dataset. Although a total of 32,203 images are included in the WIDER FACE dataset, the dataset is challenging due to large variations in scale, occlusion, pose, and background clutter. Multi-scale Detection Cascade was proposed to establish a solid baseline for the WIDER FACE dataset. A set of convolutional networks for face classification and scale classification were joined. The average precision of the easy dataset was over 60%, but none of them surpassed 75%. The performance dropped 10% for all methods on the medium dataset, and below 30% average precision for the hard dataset. The easy data set contained images and samples that were straightforward for classification. The images had distinct facial features that made it less challenging for classification models. The medium data set had a moderate level of difficulty for classification. The images in this data set may have more complexity and variability in facial features. Finally, the hard dataset was made of pictures that were particularly challenging because of factors such as occlusions, variations in lighting, or other complexities that made precise classification more difficult. The varying data subsets seemed to cause the unstable performance issue.

Finally, some researchers [10] incorporated a skin color model and face recognition using an Artificial Neural Network (ANN). The face detection and recognition algorithm was divided into four parts: skin color model, feature extractor and training images in the database using Zernike moments, processing of facial image of the training database using fuzzy sets, and face recognition using artificial neural network. ANN was used for recognizing the face from training dataset. The proposed hybrid algorithm was incorporated with the MATLAB simulator, but as seen from the previous study [7], MatLab was slow despite how hyperparameters such as MSE, Speed, recall, precision, accuracy, and elapsed time could be tuned.

III. Approaches

Considering the processing and storage limitations of our robot system, OpenCV Haar Cascade is chosen for the purpose of detecting face objects first and then VGGFace to match the target child. We will describe the two packages and compare them with other similar techniques.

A. *Face Detection using OpenCV Haar Cascade*

OpenCV's Haar Cascade was proved to be a very good solution considering our robot's constraints on storage space and the desired accuracy [11]. OpenCV's Haar Cascade is an ML object detection method that uses Haar-like features to identify certain objects from video frames or images.

Haar Cascade is known for its fast speed in real-time applications. The model is much smaller than most complex deep learning models and is appropriate for deployment on robots with limited storage. The robot has a storage limit of 64 GB and Haar Cascades typically range from 100 to 2,000 KB in size. Haar's Cascade is also of low computing complexity and allows real-time face detection even on devices with limited computing power and battery.

B. *Face Recognition Systems*

There are several different face recognition models available. A review and comparison of these models including DeepFace, VGGFace, FaceNet and OpenFace was conducted. VGGFace was selected due to its lightweight considering the limited storage and processing power of a robot and satisfactory performance.

A DeepFace [12] model is a deep learning facial recognition system from a research group at Facebook. It can have a 0.25% greater or lower performance score compared with a human's performance score of 97.35%. The convolutional neural network has 8 layers consisting of 2 convolutional layers, 1 max pooling layer, 3 locally connected layers, and 2 fully connected layers. The main problem is the long processing time and large storage required to install this package.

FaceNet [13] is a model with a 99.63% accuracy rate and it is used to generate high-quality face mapping from images. These images are from ZF-Net and Inception Network. A method called triplet loss is used as a loss function to train the model. The FaceNet model can be used as part of the classifier, or used to pre-process a face to create a face embedding which will be stored and used as input to a classifier model.

OpenFace [14] is an open source tool for computer vision. It is capable of facial landmark detection, head pose estimation, facial action unit recognition, and eye-gaze estimation. OpenFace can run in real-time with a webcam. OpenFace allows for easy integration with other software and devices through a lightweight messaging system. OpenFace has an accuracy rate of 92.92%.

VGGFace [15] refers to a series of models developed for face recognition. It was developed by the Visual Geometry Group (VGG). VGG is structured in blocks. Each block is composed of 2D Convolution and Max Pooling layers. It is available in two models, VGG16 (with 16 layers) and VGG19 (with 19 layers). As the number of layers increases within a Convolutional Neural Network (CNN), the model's ability to accommodate more complex functions increases. It is 98.78% accurate.

OpenCV's Haar Cascade Classifier plays an important role in identifying faces from live video feeds. It has shown good performance even under different angles and lighting conditions. Once a face is detected, the VGGFace Model will be invoked to determine if the detected face matches the missing children image or not. Once a successful match has been found, the robot will initiate tracking action. The VGGFace model needs to be trained with a dataset, consisting of multiple missing children images. Image augmentation techniques, including rotation, scaling, and brightness adjustment are also conducted.

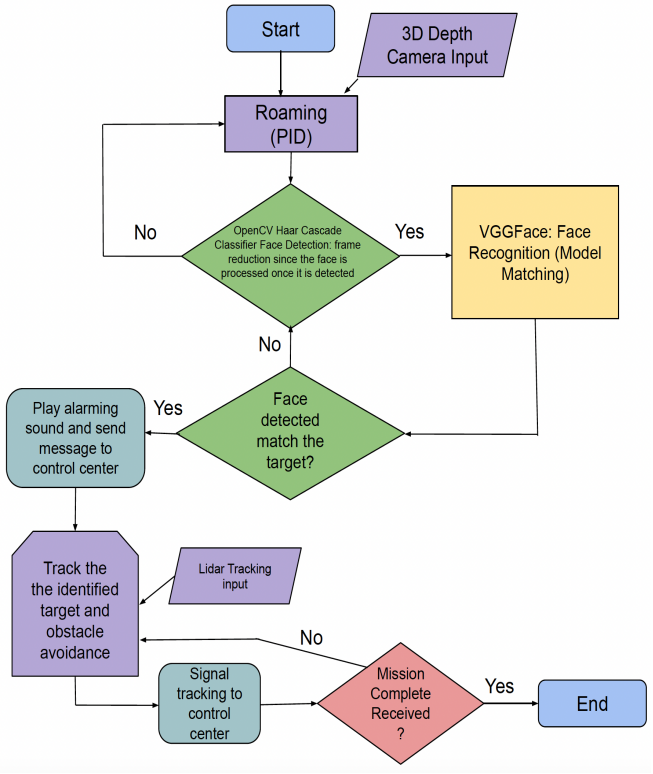

Figure 1. System Flowchart

C. *System Diagram*

As shown in Figure 1, the robot will begin its first stage: the roaming stage. It will utilize its 3D depth camera and read inputs from the unknown environment. Roaming PID (Proportional, Integral, Derivative) will assist in precise movements in unknown terrains. The OpenCV Haar Cascade Classifier will start identifying objects from the real-time image frames. It will look for faces and will drop frames without any face objects to save unnecessary processing. As seen from Figure 2, a human face is detected by Haar Cascade. If a face is detected by OpenCV Haar Cascade, VGGFace model will start the matching stage. The robot will compare key facial features from the detected image with that of the missing child. If the detected face doesn't match the target, the robot will continue roaming. If a match is found, the robot will play an alarming sound and send a message to the control center. Officials will be alerted that the target has been spotted and the robot will continue to the next stage: the tracking stage. To track and follow the target, we used the built-in KCF (Kernelized Correlation Filter) object tracking algorithm with LIDAR inputs. Users can terminate the search by sending a mission completion signal to the robot.

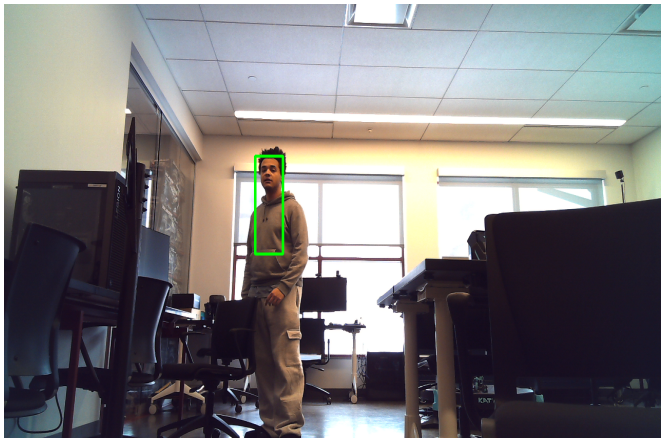

Figure 2. The robot detects human object using *OpenCV Haar Cascade*

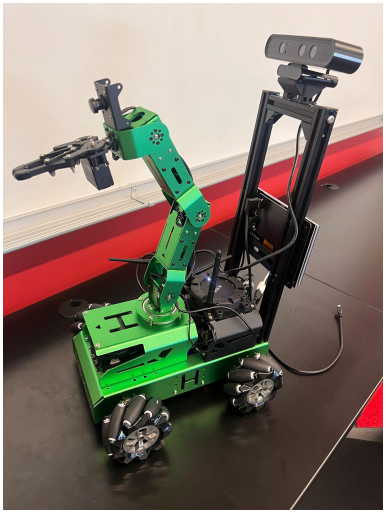

Figure 3. Yahboom x3 Jetson Nano Robot

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. Experimental Setup

To conduct our experiment, we utilized an Yahboom x3 Jetson Nano robot powered by ROS, which enables effective communication and coordination between the various components as shown in Figure 3. The robot was equipped with LiDAR and ultrasonic sensors to ensure adeptly detect and navigate around obstacles in real-time. Using its depth camera, it maps its surroundings and builds a map. The robot's movements are controlled by a PID written in Python to allow precise movements. To facilitate rapid data transmission between the Jetson Nano and the robot's control system, we chose UDP (User Datagram Protocol) for its low overhead, ensuring the swift relay of tracking data for real-time responsiveness. NVIDIA Jetson Nano is used for its potent GPU and CPU capabilities to handle the demands of real-time image processing. OpenCV, a crucial real-time image processing tool, was also enhanced by the CUDA integration on the Jetson Nano, ensuring accelerated image processing and reduced latency.

The robot is already pre-equipped with several machine learning technologies that are integral to our experiment: gesture recognition, OpenCV, and path finding libraries are several of the software packages that we used. With the large plethora of tools, we were able to create an interactive human robot application by integrating our own code with some existing Python code delivered with the robot for our application purpose.

B. VGGFace Model Training and Parameter Tuning

In the realm of facial recognition, the VGGFace model stands as a cornerstone, offering a comprehensive framework built upon deep learning architectures. This study harnesses the power of transfer learning, utilizing the pretrained VGGFace model as a foundational base. The inherent advantage of employing transfer learning lies in its ability to capitalize on the pre-existing knowledge of a model, which has been trained on a vast array of facial data. This methodology significantly truncates the training duration and data requirements, allowing for rapid deployment in the specialized context of missing child identification.

As seen from the pseudocode in Figure 4, the training regimen for adapting the VGGFace model to our specific application involves a meticulous process of fine-tuning. Initially, the pretrained model is loaded and images are augmented. The majority of the layers from the pre-trained model are kept in a 'frozen' state, thus preserving their learned weights. Subsequently, the final layers of the model were methodically 'unfrozen' and subjected to a training process, using our augmented dataset comprising images of missing children. This nuanced approach of selective training ensures the retention of the model's inherent capabilities in feature detection while customizing it to recognize the specific attributes of our target child.

```
# Main method
Function: Main():
    m = LoadPretrainedVGGFaceModel()
    ig = LoadAugmentedImages()
    m = ModelTraining(m, ig)
    return m
# Model training that handles the training process
Function: ModelTraining(m, ig): m
Input:
    m - pretrained VGGFace model
    ig - missing child augmented images
Output:
    m - retrained VGGFace model
 #freeze most layers except the last few
    for each θ ∈ m.layers[:-4] do
        θ.trainable = False
    end-for each
  #Unfreeze and train the last layers
    for each θ ∈ m.layers[:-4] do
            # Set the current θ to be trainable
            θ.trainable = True
            train each θ with epoch # and learning rate
    end-for each
return m
```

Figure 4. Pseudocode code for VGGFace training

To bolster the model's proficiency in recognizing faces under a variety of conditions, a series of image augmentation techniques were employed. These included rotations, scaling, and brightness adjustments, which collectively serve to enhance the model's ability to discern facial features across a spectrum of environmental settings. Embeddings are high-dimensional numerical vector representations of face images and designed to capture essential features of the face. Our experimental results on augmentation show that the embedding distance which measures the similarity between face images from the same child is reduced from 0.315 without augmentation to 0.212 with augmentation. The strategic use of augmentation not only diversifies the training dataset but also significantly improves the model's resilience and adaptability in real-world scenarios.

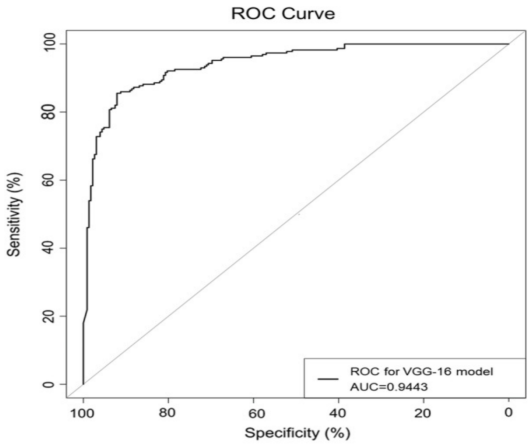

Figure 5. Receiver Operating Characteristic curve (ROC) for VGGFace model

While we are still in the stage of training and testing for a good VGGFace model, initial testing phases have shown promising results. The VGG-16 model using our students' own images and some public human face images has demonstrated a high degree of proficiency in recognizing and matching facial features from a specialized missing children's database. This is evidenced by the ROC curve depicted in Figure 5, where the model's performance is quantified by an Area under the ROC Curve (AUC) score of 0.9443, which indicates that our model has a good overall performance across all possible classification thresholds.

C. Autonomous Robot Roaming and Tracking

When the robot is roaming, a simple snake-like movement was designed, namely the robot will search its left and turn to the right in an S shape forward. If any obstacle is encountered, the robot will avoid it. The robot's speed can be set and adjusted. In the future, we plan to enable map building by utilizing the built-in Simultaneous Localization and Mapping (SLAM) system. SLAM is a robotics technique for creating a map of an unknown environment. ORB-SLAM 2 is made up of three major parallel threads: tracking, local mapping, and loop closure. The use of loop closure assists the robot in recognizing that it has returned to a previously visited location [16].

As shown in Figure 6, once the facial recognition algorithm finds the desired target face, the robot begins object tracking. The built-in OpenCV Kernelized Correlation Filter (KCF) object tracking algorithm is utilized [17]. KCF object tracking uses positive and negative space to track the identified object. It begins by extracting features from target objects and then uses the features to match the template using circular matrices. KCF then applies a kernel over the images, allowing the algorithm to detect those hard-to-capture features. It then trains a correlation filter that uses circular matrices to match and detect target images. The training algorithm uses the Fast Fourier Transform. Its speed and efficiency made it an ideal algorithm for our experiment.

V. CONCLUSIONS

In this work-in-progress experiment, we designed a robot system that can be used to locate missing children. To begin, the robot would roam in a specific designated area until it detects a human object. Then face recognition is invoked to see if there is a match with the target face. If matched, the robot will track the target while avoiding the obstacle. We have applied the transfer learning approach to quickly retrain a VGGFace model. We are still in the process of fine-tuning the hyperparameter of the model with more image augmentation techniques. It should also be noted that the robot system is not designed to operate in inclement weather such as rain or snow. This paper only considers the design of a robot system from the perspective of software and sensor integration. Outdoor weather-proof robots should be considered for this type of outdoor application. The single robot we have programmed is just a prototype that may lead to more advanced systems. It is in our interests to deploy a group of collaborative robots that can communicate and adjust their locations and tasks dynamically to speed up the search time. Within the multi-robot approach, we also need to implement efficient route planning and collision avoidance strategies that depend on local communication without centralized control for reduced communication overhead. Last but not least, the ethical issues regarding minors' photos and privacy should also be considered especially when computing applications require intensive usage of minors' data such as photos and location, etc.

Figure 6. Robot tracks human object once detected

VI. ACKNOWLEDGMENT

This work is supported in part by the National Science Foundation under Grant CNS-2117308.

REFERENCES

[1] Child Find of America: https://childfindofamerica .org /resources /facts - and-stats-missing-children/

[2] AI for missing and exploited Children: https://siliconangle.com/2017/03/13/ai-became-key-technology-finding-missing-exploited-children-sxsw/

[3] P. S. Chandran, N. B. Byju, R. U. Deepak, K. N. Nishakumari, P Devanand, and P. M. Sasi, "Missing Child Identification System Using Deep Learning and Multiclass SVM", *2018 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*

[4] M. Naga Sravani, and S. Kumar Ghosh "Implementation of Children ActivityTracking System Based on Internet of Things", *Proceedings of the Third International Conference on Computational Intelligence and* Informatics.

[5] A. Tsanousa,V.R. Xefteris, D. Ntioudis, C. Chatzigeorgiou, G. Meditskos, S. Vrochidis, C. Z. Patrikakis, and I. Kompatsiaris, "Localization module for missing child scenario in IoT safety domains", *2021 IEEE International Conference on Consumer Electronics (ICCE)*

[6] The British Columbia Institute of Technology (BCIT): "Search and Rescue: Real-time Image Detection." *BCIT, May 2022*, www.bcit.ca /learning-teaching- centre/ remotely -pilo ted-aircraft -systems / industry - collaboration /real - time - image - detection - for - search-and-rescue/

[7] K. H. Teoh et al. "Face recognition and identification using deep learning approach", *Journal of Physics: Conference Series,* vol. 1755, no. 1, 2021, p. 012006, https://doi.org/10.1088/1742-6596/1755/1/012006.

[8] H. Shivalila et al."Face detection and recognition using face mesh and deep neural network", *Procedia Computer Science*, vol. 218, 2023, pp. 741–749, https://doi.org/10.1016/j.procs.2023.01.054.

[9] S. Yang et al. "Wider face: A face detection benchmark ", *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR*), 2016, https://doi.org/10.1109/cvpr.2016.596.

[10] R. P. Singh1 and D. Kaur, "Evaluation of Performance of the Face Detection Using Skin Colour Model and Face Recognition Using ANN", *International Journal of Computer Science and Mobile Computing ,* Vol.4 Issue.9, September- 2015, pg. 237-241

[11] OpenCV Cascade classifier: https://docs.opencv.org/4.x/db/d28/tutorial_cascade_classifier.html

[12] S. Sefik "Face Recognition with Facebook Deepface in Keras." Jan. 2022, https://sefiks.com/2020/02/17/face-recognition-with-facebook-deepface-in-keras/#google_vignette

[13] S. Florian et al. "FaceNet: A unified embedding for face recognition and clustering." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 2015, https://doi.Org/10.1109/cvpr. 2015.7298682.

[14] T. Baltrušaitis, P. Robinson and L. P. Morency, "OpenFace: An open source facial behavior analysis toolkit," *2016 IEEE Winter Conference on Applications of Computer Vision (WACV),* Lake Placid, NY, USA, 2016, pp. 1-10, doi: 10.1109/WACV.2016.7477553.

[15] Q. Cao et al. "VGGFACE2: A dataset for recognising faces across pose and age," *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition* (FG 2018), 2018, https://doi.org/10.1109/fg.2018.00020.

[16] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, Oct. 2017, doi: 10.1109/TRO.2017.2705103.

[17] J. F. Henriques, R. Caseiro, P. Martins and J. Batista, "High-Speed Tracking with Kernelized Correlation Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 3, pp. 583-596*, 1 March 2015, doi: 10.1109/TPAMI.2014.2345390.