Smart Infant Monitoring System Using Computer Vision and AI

Gurpreet Singh, Abhishek Raj Shekhar, Xinrui Yu, and Jafar Saniie
Embedded Computing and Signal Processing Research Laboratory (http://ecasp.ece.iit.edu/)
Department of Electrical and Computer Engineering
Illinois Institute of Technology, Chicago, IL, U.S.A.

***Abstract*—The new era of technology is being greatly influenced by the field of artificial intelligence. Computer vision and deep learning have become increasingly important due to their ability to process vast amounts of data and provide insights and solutions in a variety of fields. Computer vision, deep learning and signal analysis have been used in a growing number of applications and services including smart devices, image, and speech recognition, healthcare, etc., one such device is an infant monitoring system. It monitors the daily activities of the infant such as their sleeping patterns, sounds, and movements. In this paper, deep learning and computer vision libraries were used to develop algorithms to detect whether the infant was in any uncomfortable situation such as sleeping on its back, face being covered and whether the infant was awake. The smart infant monitoring system detects the infant's unsafe resting situation in real time and sent immediate alerts to the caretaker's device. This paper presents the design flow of a smart infant monitoring system consisting of a night vision camera, a Jetson Nano, and a Wi-Fi internet connection. The pose estimation and awake detection algorithms were developed and tested successfully for different infant resting/sleeping situations. The smart infant monitoring system provides significant benefits for safety and an improved understanding of infants' sleep patterns and behavior.**

Keywords—infant monitoring system, deep learning, face detection, pose detection, jetson nano, computer vision.

I. INTRODUCTION

This project is aimed at developing a smart infant monitoring system that utilizes computer vision technology to automatically recognize the potentially harmful conditions an infant can be in at any instance. After the detection of any anomalies by the algorithms, alerts can be sent to the caretaker's devices, if their intervention is necessary for the infant's well-being.

Current infant monitoring system transfer video and audio cues to the caretakers. The full scale of insights that can be derived from this data is not fully explored. This project proposes a prototype addressing these basic cases: (a) Determine instances whenever the infant is sleeping on its stomach with face down on the mattress. (b) Determine when the infant's face is covered with a blanket, or the body is not covered with a blanket. (c) Determine whether the infant is awake during the time of nap. Cases (a) and (b) solutions involve building pose estimation models and tweaking them to cater to these specific problem statements and case (c) involves facial landmark detections. Deep learning-based algorithms are used to achieve the required results. For the real-time application of the developed prototype algorithms, the NVIDIA Jetson Nano module is used. This is an embedded device packing a CPU and dedicated GPU. Hence it has the optimal processing power to build a real-time system, run neural networks and at the same time keep the latency minimum. The significant needs of such systems can be justified using the following points:

- Annually, there are around 3,400 SUIDs (sudden unexpected infant deaths) in the United States [1]. These infant fatalities, which affect those under a year old, are not immediately apparent. SUIDs can be classified into the following categories: sudden infant death syndrome (SIDS), unknown origin, accidental bed strangulation, and suffocation. In 2020, around 1,389 infants died from SIDS, 1,062 from unclear causes, and 905 from accidental suffocation and strangulation in bed.
- Generally, the safest position for an infant to sleep is on their back [2]. Infants may squirm to their sides in sleep, they can be in this posture without risk as they get stronger. However, lying on one's stomach, especially at first, increases the risk of asphyxia and SIDS. Sleeping face down on the mattress could be harmful to an infant. The formation of pockets of carbon dioxide in the space where the infant lays can be lethal if the infant is sleeping on its stomach face down. As the infant re-breathes the air exhaled, there is a drop in the oxygen level and a rise in the levels of carbon dioxide. And with oxygen suffocation is caused that may lead to fatality. This deprivation of oxygen can be the reason for the development of abnormality which can be directly linked to SIDS [2].
- Infants wake up at nightfall due to discomfort or disturbed circadian rhythm. An infant's erratic sleep patterns may mean that no one in the family snoozes very soundly. Parents prefer to know when their infant is awake which can be made possible by the proposed smart infant monitoring system.
- Studies prove that over 60% of parents with babies under the age of 24 months get no more than three and a half hours of sleep every night. Long-term insomnia can be developed due to disturbance in sleep patterns in new parents. Mental health problems are common in such cases leading to chronic anxiety and depression. Physical health problems such as heart disease, diabetes, and obesity can be caused due to such a lifestyle [3].
- The proposed system will help the new parents to know the situations when the infant needs care, eliminating the

need for constant attention hence significantly reducing their stress levels and letting them have a healthy sleep.

II. Technical Description

The proposed prototype's performance was evaluated on both a traditional personal computer GPU and the NVIDIA Jetson Nano developer kit [4]. NVIDIA Jetson Nano comes with a built-in Maxwell 128-core GPU, which can run multiple neural networks simultaneously for computer vision tasks like image processing, speech recognition, object detection, etc.

A. Hardware

The NVIDIA® Jetson Nano development kit is the processing unit. It's the ideal choice for edge computing and real-time neural network applications since it's a single embedded platform tailored specifically for image processing and neural networks. It has a Quad-core ARM A57 @ 1.43 GHz CPU, a 128-core Maxwell graphics processing unit (GPU), 4 GB 64-bit LPDDR4 25.6 GB/s memory, four USB 3.0, USB 2.0, Micro-B ports, microSD storage (up to 1TB), 2x MIPI CSI-2 DPHY lanes camera, Gigabit Ethernet M.2 Key E connectivity, HDMI and display port, mechanical 69 mm x 45 mm, 260-pin edge connector, etc., A USB interface connects a night-vision camera to the Jetson Nano. The camera has 5 Megapixels sensor and supports 1080P resolution. It has photosensitive resistance and an IR-CUT built-in, allowing it to detect and identify light to automatically transition between night-vision and day-time photography modes. A Wi-Fi adapter is linked to the Jetson Nano's USB port for wireless internet connection.

B. Software

Linux4Tegra (L4T) operating system (OS), was installed on the Jetson Nano board's SD card (32 GB). All of the necessary packages, such as OpenCV, Dlib, and OpenPose, were installed on Jetson Nano which came with preloaded rudimentary models. For pose estimation, different models were tested such as TRTPose [5], MediaPipe [6], Openpose [7], etc. OpenPose is found to be best suited for the proposed system. One of the libraries used in the awake detection algorithm is the Dlib image processing library. It is a deep learning-based C++ library used for image processing, threading, networking, etc. Dlib is a license-free open-source library that can run across multiple platforms. This library is also present in python. Python programming language is versatile and easy to use. It also has a vast number of libraries and is platform-independent. This makes python quite convenient. Therefore, the suggested system's source code is written in Python. The Jupyter notebook, a web-based IDE is used as our working environment.

III. Methodology

The smart infant monitoring system comprises a night vision indoor webcam, an NVIDIA Jetson Nano module, a Wi-Fi adapter, and a monitor for displaying alerts. The infant is monitored in real time by a night vision camera which sends the input feed to the Jetson nano via a Wi-Fi server [8]. The camera is set up such that the field of view of the camera covers the infant perfectly from head to toe. This input feed is processed in the Jetson Nano where image processing-based pose estimation and awake detection algorithms are implemented simultaneously. Further on, the results obtained from these algorithms are analyzed and alerts are sent out to parents or caregivers accordingly as shown in Fig. 1.

There are two integral parts to this problem. To detect whether the infant is in a vulnerable position, estimating the pose and body parts becomes the most important part of the problem. By first estimating the pose by utilizing Artificial Intelligence, we can write basic algorithms on top of it to get the desired result. For instance, we can get the instances when the infant is not covered in a blanket by detection of the lower body parts like hips and leg joints. Some positions when the infant is in critical conditions like sleeping with a 'face down in the mattress' case can also be detected by detecting the nose of the infant. Hence it all boils down to accurately estimating the pose by detection of different body parts. Secondly, to detect when the infant is awake, we opt for localized body part detections. Facial landmarks are detected by pre-trained models and the coordinates of respective landmarks are grouped to detect the eyes. After the detection of the eyes, formulas can be constructed on top of it to get the instance when the eyes are closed or opened for a specified time duration. Based on this, conclusions can be made about when the infant is awake or asleep. Now utilizing Jetson Nano these model algorithms can be implemented on the video stream captured by the camera in parallel to achieve the objectives of the project.

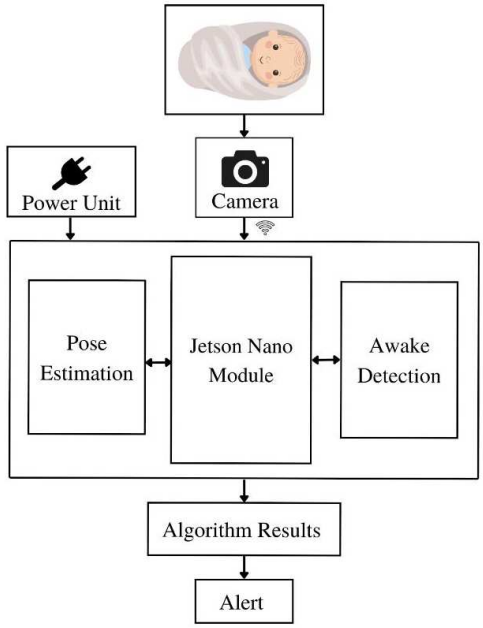

Fig. 1. Flowchart demonstration of the process for the development of the prototype Infant Monitoring System.

IV. Pose Estimation

Pose estimation has received a significant amount of interest in the discipline of computer vision. The capacity to employ computer vision algorithms to recognize and follow the motion of a person or an item in real time is increasing in popularity since it has a wide range of applications across industries [9]. Pose estimate has emerged as a valuable tool in graphics, medical rehabilitation, gaming, sports biomechanics, automation, and surveillance in the ever-evolving technological

era. technology. Because stance movements are frequently driven by unique human activities, knowing a human's body position is crucial for action identification. Pose estimation, in essence, anticipates possible positions in a picture or video based on a person's body components and joint positioning.

A. Types and Challenges

Pose estimation can be segregated into two categories 2 Dimensional and 3 Dimensional. In 2D human pose estimation, from visuals like images and videos, the 2D position or spatial placement of the human body's essential points is recognized. Traditional 2D human pose estimate methods employ feature extraction techniques for each body part and treat human bodies as stick figures to get global posture structures [10]. In contrast, 3D Human Pose Estimation is utilized to predict where body joints would be in 3D space. The 3D human mesh may also be recovered from visuals like images or videos using 3D pose techniques [11]. For estimation of the pose of the infant in a cradle, we hereby require the 2D pose estimation techniques as the information needed is the spatial location of body key points from the video feed to run the algorithm we want.

Position estimation is a difficult task since the body's appearance dynamically varies due to various types of clothing, arbitrary occlusion, viewing angle, and surrounding environments [7]. Human pose estimation must be resistant to real-world variables such as illumination and weather. As a result, image processing methods struggle to recognize fine-grained joint coordinates. Tracking tiny and faintly visible joints is especially challenging.

Some of the most popular models currently developed in the 2D pose estimation space are OpenPose [12], Movenet [13], Posenet [14], DCPose [15], DensePose [16], HigherHRNet [17], AlphaPose [18], TransPose [19], etc. Also, there is a model called trt_pose which aims to enable real-time pose estimation on NVIDIA platforms. This model is an open-source NVIDIA project which is optimized for use with the NVIDIA Jetson processor. Any of these models is capable of mapping the key points of a person and estimating the corresponding poses even in occluded scenarios. OpenPose is the model utilized in this project to perform pose estimation.

B. Architecture and Algorithms

OpenPose follows a bottom-up approach i.e., identifying the key points first and assigning that to the individuals in the post-processing phase of the model [7]. The pipeline is a coalescence of different advanced fields such as calculus, set theory, graph theory, and deep learning. The first step followed in this pipeline is preprocessing of input frame images. The image is converted from [0, 255] to [-1, 1] [20]. To extract the features from the image, the frames are passed through the image extraction layer VGG-19 [7]. Then the network is split into two branches to predict different things. A set of 18 different confidence heat maps representing each different part of the human body is predicted by the first branch [7]. The location of different body parts out of 18 generated matrices is extracted from the information derived. The confidence heatmap is then transformed into certainty by using non-maximum suppression (NMS) [20]. After passing through NMS, the non-zero-pixel values now denote the candidates for each one of the body parts.

After this, the graph theory is applied to obtain a complete bipartite graph where the edges denote each possible association, and the vertices represent part candidates. Now the objective is to assign weights to each edge of the graph to find the right connection. This is where the significance of the second layer is, it gives the sense of which two parts can be combined or association between a couple of parts can be made to give a pair. This is done by producing 38-part affinity field matrices in association with every pair [7]. This part affinity field matrix gives information on the position and direction in each x and y-axis of pairs. There are multiple stages of the networks where the output of the previous stage is refined. The line integral of PAF and the bipartite graph is taken to assign the weights to the graph edges. A line integral is integral that evaluates the function to be integrated along a curve. In practical terms, it defines the effect of any field.

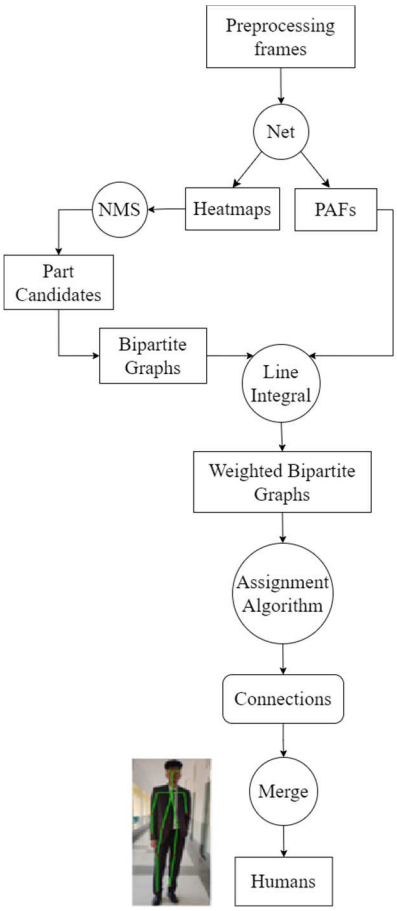

Fig. 2. OpenPose Pipeline for Pose Estimation.

The objective then is to find the connection that gives a maximum total score. This algorithm is called the assignment algorithm. In this sorting is done from maximum to minimum, and a greedy approach is used to find the maximum of each pair of connections [20]. These detected connections are then to be transformed into human skeletons. It is considered that each connection belongs to different humans. At this point, merging is done to club all the connections having the same part coordinates and part index values to form an individual human [7]. The output is hence obtained where each human figure is

defined as a set of parts containing its index, coordinates, and score.

Now the aim is to determine the conditions in which the infant is in a vulnerable state. Algorithms are designed using the principle that if the infant's nose is not in view for a while, a conclusion can be made either the infant is sleeping in a bad posture on its stomach, or the blanket is covering the infant's face both of which are not healthy situations for an infant to be in [8]. These specific body parts can be detected by accessing the part indices in the COCO format [21]. If hips, knees, and feet are detected then it can be concluded that the blanket is not completely on the infant.

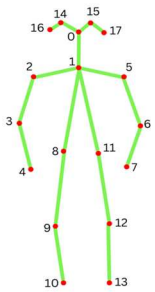

Fig. 3. Pose Output Format (COCO) [21].

C. Results

Along with its real-time model based on OpenPose, the suggested pose estimation approach was tested on several stock infant images, as shown in Fig. 4. The suggested approaches can successfully estimate the poses and algorithms are executed on those poses to generate alerts. In Fig. 4(a), the algorithm is not able to detect the nose of the infant, this is a condition when alerts will be issued as the infant's posture is not a healthy one. Other images in Fig. 4 are relatively healthy positions for an infant to sleep in.

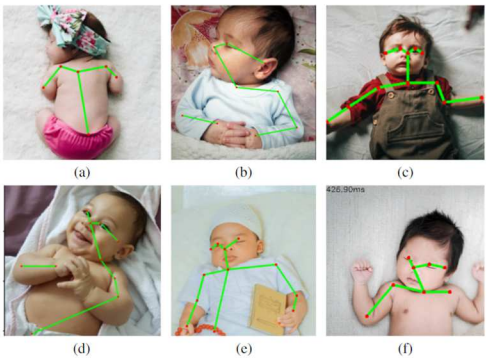

Fig. 4. Results obtained when pose estimation is performed on infants.

V. Awake Detection

To figure out whether the infant is awake or not, an awake detection algorithm has been created. It uses two levels-facial feature extraction from the input frame and calculation of the eye aspect ratio (EAR) values. The night vision camera provides the real feed of the baby. The feed is sent to the Jetson Nano module via a Wi-Fi connection. The algorithm present in the jetson nano is implemented on each frame of the input video feed. There are two proposed methods for facial feature detection: OpenCV Haar cascades and Dlib face detection. The Haar cascade algorithm selects features based on the difference in the sum of intensities of the dark and light regions in the Haar-like feature. An integral image is created where a pixel in the image is equal to the sum of the pixels to its left and above. The Adaboost method generates a robust classifier by linearly combining weak classifiers [22]. Finally, a cascade classifier is formed by chaining together more complicated classifiers such as Adaboost together.

Dlib face detection also has two different methods for facial feature extraction and detection. The histogram of oriented gradients with linear support vector machine (HoG+SVM) algorithms is capable of implementing a very efficient facial recognition system [23]. HoG is a reductive yet fast algorithm that deconstructs a picture into constituent groups of pixels from which it extracts features that may be connected to recognized categories of objects. To do so, it first generates a low-level histogram that defines the contours of objects in the picture based on pixel intensity and the extent to which it abruptly goes off. The linear SVM classifiers are trained to classify the input images and the resultant output is obtained. Max-Margin (MMOD) CNN algorithm is a strong and dependable GPU-accelerated face detector that uses a convolutional neural network (CNN) and is significantly more capable of collecting faces from obscure angles and in difficult settings, making it suitable for casual surveillance and urban analysis [23].

A. Architecture and Algorithm

Dlib is a powerful and widely used facial recognition library that offers a perfect mix of resource utilization, latency, and accuracy, making it suitable for real-time face recognition [24]. It's becoming a frequent, if not necessary, library in the facial recognition scene, and it's a better fit for our computer awake detection framework. For awake detection, Dlib, an image processing package, is utilized to extract the facial landmarks from each frame using the Dlib pre-trained facial landmark detector, which is then used for awake detection [25]. The histograms are formed using the frequencies of the gradient direction of the face. The pre-trained face detector function is used to extract all 68 facial landmarks [26]. The eyes feature 38 to 42 for the left eye and 43 to 48 for the right eye is extracted and represented in Fig. 5 [26]. The eye-aspect ratio (EAR) is calculated for both eyes.

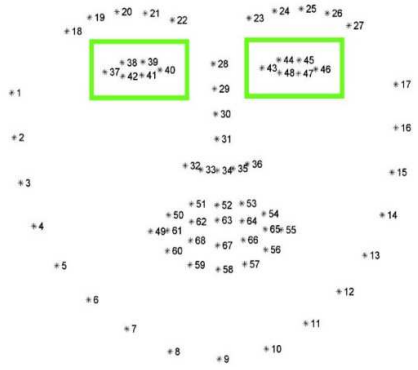

Fig. 5. Facial Landmark points detected by Dlib [27].

The points marked in Fig. 6 are the eye coordinates. The EAR for an open eye is between 0.2- 0.4 and between 0.1-0.15 for a closed eye as shown in Fig. 7. The EAR of both eyes is calculated using the eye-aspect ratio equation shown in (1).

$$EAR = \frac{|P2 - P6| + |P3 - P5|}{2(|P1 - P4|)} \quad (1)$$

$$Average\ EAR = \frac{EAR\ left + EAR\ right}{2} \quad (2)$$

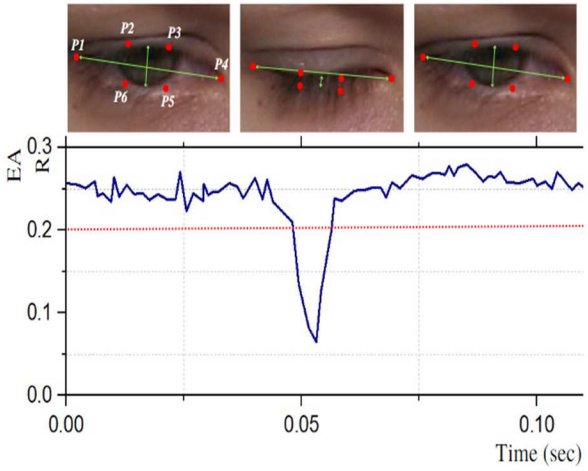

Fig. 6. Demonstration of eye aspect ratio variation over time

The average distance between the upper and lower eyelids is divided by the eye size which is determined by the horizontal endpoints of the eye. The average value is taken for both eyes. The points marked in Fig. 6 are the eye coordinates. The EAR for an open eye is between 0.2- 0.4 and between 0.1-0.15 for a closed eye as shown in Fig. 7. The EAR of both eyes is calculated using the eye-aspect ratio equation shown in (1). The average distance between the upper and lower eyelids is divided by the eye size which is determined by the horizontal endpoints of the eye. The average value is taken for both eyes [28]. The algorithm calculates the average eye aspect ratio as shown in (2). If the values exceed a threshold value (0.2 in our case) for a certain number of consecutive frames determined by a consecutive frame threshold, it is concluded that the infant is awake. The counter variable is initialized which is incremented by one for each frame where the eyes were open. The counter value is then compared with the frame threshold set up to be 200 for the algorithm. If the value of the counter variable exceeds the set threshold, a notification is delivered to the parent that their child is awake and in the need of assistance [29]. However, if the counter value is below the threshold, it indicates that the infant is sleeping. The block design of our suggested awake detection algorithm is shown in Fig. 7.

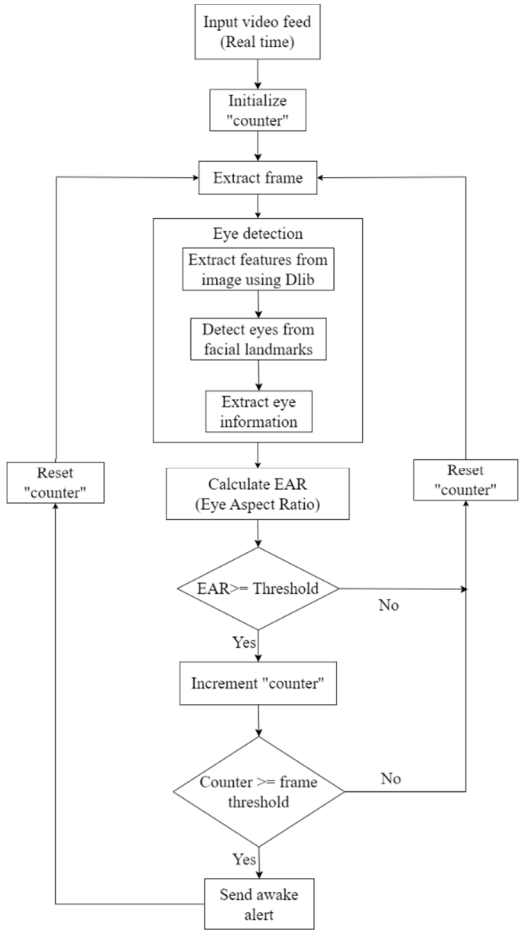

Fig. 7. Awake Detection Algorithm.

B. *Results*

Along with its real-time model, the suggested detection approach was tested on several infant images, as shown in Fig. 8. The suggested approaches can successfully detect the eye-closed condition shown in Fig. 8(a) and 8(c), as well as the awake condition shown in Fig. 8(b) and 8(d).

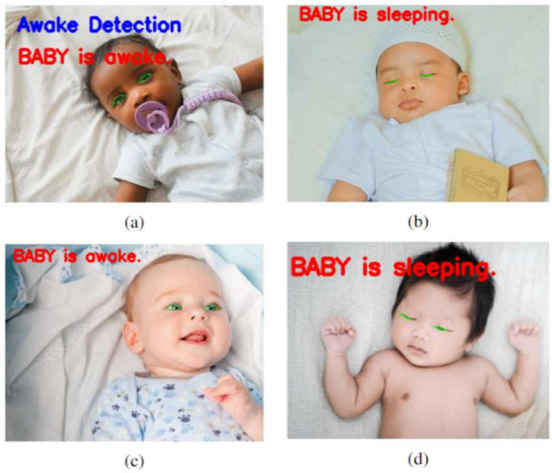

Fig. 8. Result for awake detection. (a) Baby is awake (EAR=0.29), (b) Baby sleeping (EAR=0.11), (c) Baby is awake (EAR=0.35), (d) Baby is sleeping (EAR =0.14).

The Dlib face detector was used for awake detection. Another alternative method for awake detection was using the MTCNN [30] model for eye detection and using a queue-based EAR value storage and sorting system instead of using a single variable (counter) for a more elaborate and efficient algorithm.

VI. Conclusion

This paper presents a smart infant monitoring system based on technologies such as computer vision and artificial intelligence. The proposed prototype works on the principles of image processing and deep learning to detect certain movements of infants and predict instances whenever the infant is in a vulnerable situation. Then the corresponding alerts are issued to the caretaker or parents. A microcontroller-based system capable of real-time applications and an interface to a camera are used to implement the algorithms. This monitoring system can find great application in the home environment, daycare centers, and hospitals, especially in neonatal intensive care units (NICUs). By providing continuous monitoring of an infant, this system can help provide vital information to medical staff, improving the efficiency and the service provided by the caretakers. With this infant monitoring system, single parents or working parents can carry out their daily chores while still being able to keep an eye on their infant.

References

[1] "Data and statistics for SIDS and suid," *Centers for Disease Control and Prevention*, 21-Jun-2022. [Online]. Available: https://www.cdc.gov/sids/data.htm/. [Accessed: 01-Mar-2023].

[2] L. Anderson and L. Anderson, "My baby sleeps face down in the mattress, should I worry?," *MomInformed*. [Online]. Available: https://mominformed.com/my-baby-sleeps-face-down-in-the-mattress-should-i-worry/. [Accessed: 01-Mar-2023].

[3] "New parents have 6 months sleep deficit during first 24 months of Baby's Life," *Medical News Today*. [Online]. Available: https://www.medicalnewstoday.com/articles/195821#1. [Accessed: 01-Mar-2023].

[4] "Jetson Nano Developer Kit," *NVIDIA Developer*, 28-Sep-2022. [Online]. Available: https://developer.nvidia.com/embedded/jetson-nano-developer-kit. [Accessed: 01-Mar-2023]. M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[5] Nvidia-Ai-Iot, "Nvidia-ai-IOT/TRT_POSE: Real-time pose estimation accelerated with Nvidia TENSORRT," *GitHub*. [Online]. Available: https://github.com/NVIDIA-AI-IOT/trt_pose. [Accessed: 01-Mar-2023].

[6] "Mediapipe," *PyPI*. [Online]. Available: https://pypi.org/project/mediapipe/. [Accessed: 01-Mar-2023].

[7] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2D pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2021.

[8] T. Khan, "An intelligent baby monitor with automatic sleeping posture detection and notification," *AI*, vol. 2, no. 2, pp. 290–306, 2021.

[9] S. C. Babu, "A 2019 guide to human pose estimation with deep learning," *Nanonets AI & Machine Learning Blog*, 05-Aug-2019. [Online]. Available: https://nanonets.com/blog/human-pose-estimation-2d-guide/. [Accessed: 01-Mar-2023].

[10] E. Odemakinde, "Human pose estimation with deep learning - ultimate overview in 2023," *viso.ai*, 25-Feb-2023. [Online]. Available: https://viso.ai/deep-learning/pose-estimation-ultimate-overview/. [Accessed: 01-Mar-2023].

[11] A. Benzine, B. Luvison, Q. C. Pham, and C. Achard, "Deep, robust and single shot 3D multi-person human pose estimation from monocular images," *2019 IEEE International Conference on Image Processing (ICIP)*, 2019.

[12] CMU-Perceptual-Computing-Lab, "CMU-Perceptual-Computing-Lab/openpose: OpenPose: Real-time multi-person keypoint detection library for body, face, hands, and foot estimation," *GitHub*. [Online]. Available: https://github.com/CMU-Perceptual-Computing-Lab/openpose. [Accessed: 01-Mar-2023].

[13] "MoveNet: Ultra fast and accurate pose detection model. : tensorflow hub," *TensorFlow*. [Online]. Available: https://www.tensorflow.org/hub/tutorials/movenet. [Accessed: 01-Mar-2023].

[14] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A convolutional network for real-time 6-DOF camera relocalization," *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[15] Z. Liu, H. Chen, R. Feng, S. Wu, S. Ji, B. Yang, and X. Wang, "Deep dual consecutive network for human pose estimation," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021

[16] R. A. Guler, N. Neverova, and I. Kokkinos, "DensePose: Dense human pose estimation in the wild," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[17] B. Cheng, B. Xiao, J. Wang, H. Shi, T. S. Huang, and L. Zhang, "Higherhrnet: Scale-aware representation learning for bottom-up human pose estimation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[18] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu, "Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–17, 2022.

[19] S. Yang, Z. Quan, M. Nie, and W. Yang, "Transpose: Keypoint localization via Transformer," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[20] A. Solano, "Human pose estimation using openpose with tensorflow (part 2)," *Medium*, 20-Nov-2017. [Online]. Available: https://arvrjourney.com/human-pose-estimation-using-openpose-with-tensorflow-part-2-e78ab9104fc8. [Accessed: 01-Mar-2023].

[21] "Common objects in context," *COCO*. [Online]. Available: https://cocodataset.org/#home. [Accessed: 01-Mar-2023].

[22] Z. Mahmood, T. Ali, and S. Khattak, "Automatic player detection and recognition in images using AdaBoost," *Proceedings of 2012 9th International Bhurban Conference on Applied Sciences & Technology (IBCAST)*, 2012

[23] A. Rosebrock, "Face detection with dlib (Hog and CNN)," *PyImageSearch*, 17-Apr-2021. [Online]. Available: https://pyimagesearch.com/2021/04/19/face-detection-with-dlib-hog-and-cnn/. [Accessed: 01-Mar-2023].

[24] *dlib C++ Library - Machine Learning*. [Online]. Available: http://dlib.net/ml.html. [Accessed: 01-Mar-2023].

[25] *dlib C++ Library - Image Processing*. [Online]. Available: http://dlib.net/imaging.html#shape_predictor. [Accessed: 01-Mar-2023].

[26] A. Rosebrock, "Facial landmarks with dlib, opencv, and python," *PyImageSearch*, 03-Jul-2021. [Online]. Available: https://pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/. [Accessed: 01-Mar-2023].

[27] *i·bug - resources - Facial point annotations*. [Online]. Available: https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/. [Accessed: 01-Mar-2023].

[28] S. Shaily, S. Krishnan, S. Natarajan, and S. P., "Smart Driver Monitoring System using AI," *Advances in Medical Technologies and Clinical Practice*, pp. 225–250, 2021.

[29] A. Rosebrock, "Eye blink detection with opencv, python, and dlib," *PyImageSearch*, 20-Feb-2023. [Online]. Available: https://pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/. [Accessed: 01-Mar-2023].

[30] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.