

Nekray: A Linux Kernel-Based Customized Operating System for Information Kiosk


A. S. M. Mehedi Hasan Sad, American International University, Bangladesh

Md Mashrur Sakib Choyon, American International University, Bangladesh

Abu Hasnat Md Rhydwan, American International University, Bangladesh

Kawshik Shikder, American International University, Bangladesh

Chowdhury Akram Hossain, American International University, Bangladesh

 <https://orcid.org/0000-0002-8769-2833>

ABSTRACT

In recent years, the demand of kiosk devices has increased significantly for relaying information in organizations, institutions, or any other service centers. They have become a better alternative for traditional human assistance or reception desks. However, there are no dynamic operating systems or user interfaces available for kiosk devices. This paper represents a development of an operating system for kiosk devices called ‘Nekray’, which was built on Linux Kernel environment. It was designed to be dynamic, fast, user-friendly, and user interactive. The developed operating system avails the option to change the data of its features as per requirement. It also supports plug and play feature and can be installed in any low-cost hardware board. Furthermore, built-in AI is also a part of the developed system that performs its features through image processing. The system maintains the privacy and interactive transition of data to its users on kiosk devices.

KEYWORDS

3D Map, 3D Seat Plan, Computer Vision, Dynamic System, Embedded Multiple Sensors, Embedded System, Image Processing, Plug and Play, Smart System

INTRODUCTION

Implementation of smart automated systems are gradually increasing globally. Some of the applications of these devices include providing rapid information, automated healthcare facility, and home automation. The rising global population has also forced different organizations to move towards automated systems to serve their users. Among these devices, the kiosk stands out due to its usability and capability to deliver data without any external help (Sad et al., 2020).

The kiosk is an automated electric device that uses different modern hardware and software to ensure a wide range of features such as wayfinding, advertising, hospitality, and health check-in (Joshi et al., 2017). Currently, they are used almost everywhere starting from schools to large offices to ensure a seamless user experience without the need for any sort of human assistance. As reported in the literatures, kiosks have seen their application in markets (Rajendran, 2018), universities (Ekşioğlu et al., 2018), restaurants (Han et al., 2019), and healthcare (Afzali et al., 2017; Gladia & Kavya, 2017). In addition to that, researchers are also assessing the usability of the kiosks for virtual reality

DOI: 10.4018/IJERTCS.2021070104

Copyright © 2021, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

development (Graham et al., 2018) and laboratories (Branzila et al., 2018). Another type of kiosk that is widely used in different organizations to rectify their operation is known as the information kiosk.

An operating system is a major part of every information kiosk as they control the overall working of the system and a user-friendly operating system can easily improve user interaction, which is the primary objective of any kiosk. In the case of an information kiosk, a fast operation is necessary as most of the time they are placed in crowded areas. So, choosing an appropriate operating system can be challenging. In this research, a Linux kernel based operating system was developed that can handle the challenges in any information kiosk. The hardware requirements of the OS have been kept low so that it can be installed almost on every system. The key contributions of this research are:

- Development of a Linux kernel based operating system for information kiosk.
- Implementation of artificial intelligence (AI) within the operating system.
- Improving hardware resource utilization for kiosk operating system to ensure constant performance.
- Development of a responsive UI that changes based on user.

The proposed solution mainly focuses on the user experience and user's comfort in using a kiosk. The developed OS eliminates the unnecessary background process in the system and thus, making the device faster in fetching or delivering data. The dynamicity of the system enables to design the UI as per the individual requirements of any organization. Furthermore, the combination of embedded systems with multiple sensors enables the device with more features. The combination of AI and attractive UI makes the system more unique compared to the available kiosk devices found in the market. The OS also utilizes power more efficiently by reducing wastage of power by the device. Programs like 3D visualization of an organization's area or building significantly increase the users' satisfaction in using the Nekray installed kiosk devices. With a quite low specifications requirement, the Nekray OS enables the advantage of installing within a much cheaper and widely available microcontroller like the raspberry pi. Therefore, the proposed solution can be used in almost every organization in kiosk devices with a much lower installation cost and having greater advantages on multiple aspects.

The paper is organized as follows. The "Related Research Work" section highlights the recent developments in the field of IoT and illustrates different applications of the Linux operating system. The development of the operating system has been shown in the "OS Architecture" section. The "Custom Feature Development" section showcases different features that were developed for an information kiosk to run on the developed operating system. Various operations and features of the OS have been shown in the "Results" section. Finally, a summary of the overall research has been provided in the "Conclusion" section.

RELATED RESEARCH WORK

With the continuous growth of IoT devices, numerous researchers have emphasized developing more user-friendly and efficient operating systems for these devices. As a result, there has been an unprecedented trend for developing application architectures and frameworks recently. In (Kum et al., 2017), the authors proposed an architecture for IoT applications, which uses the Fog computing concept to perform different services by the IoT applications. In another work, various operating systems for IoT devices have been investigated by the authors to help the researchers working in this field to choose the suitable ones for their applications and the work also summarizes the features of different software and hardware platforms (Afzal et al., 2019). In (Choyon et al., 2020), an IoT based health monitoring device was developed using raspberry pi and necessary sensors. In (Payne & Abegaz, 2018), best practices to secure IoT based devices from different types of cyber-attack have been analyzed. In another similar type of research, a comparison between different latest OS has

been shown in terms of their architecture, memory, efficiency, and real-time performance (Sabri et al., 2017). The potential of the reverse engineering framework to find the data vulnerabilities within Linux-based OS has been also investigated. The authors of that research have successfully used their developed framework to reverse-engineered the WeMo smart plug communication protocol to analyze its network traffic to detect and remove any sort of design flaws within that system (Liu et al., 2020).

Linux has undergone a tremendous leap recently and has become one of the widely used OS, especially for IoT devices (Serra et al., 2020). In (Vishnubhatla, 2020), a Linux server-based ticketing kiosk has been demonstrated that was developed using a processor, Qtopia based GUI and MYSQL based database server. In another application, Linux kernel-based virtual machine hypercall has been used to develop and implement a virtual machine control scheme that can significantly improve the efficiency of the host machine by reducing CPU loads (Choi & Hong, 2019). In (Boras et al., 2020), a performance evaluation has been conducted on three Linux distributions. In order to measure performance, the authors consecutively installed the distributions on the same desktop and the result shows that the Pop!_OS 20.4 Linux distribution performed the best. ARM architecture-based devices mostly use the customized version of different Linux distributions. In (Swain et al., 2020), the authors also performed a comparative analysis among embedded firmware of IoT devices. On the other hand, Zhang et al. (2017) characterized different Linux kernel versions by their properties and proposed a methodology to identify different kernel struct layouts.

OS ARCHITECTURE

System Architecture

Existing operating systems such as Windows and Mac are prone to slow operation due to their heavy architecture and it is not preferable for a kiosk operating system. To solve that issues, Nekray was developed based on the Linux kernel, which is considerably faster than other operating systems (Gerofi et al., 2018). The development of an operating system depends on various aspects including hardware, kernel, and applications as shown in Figure 1. Before the development of the OS, hardware requirements need to be determined to ensure proper performance. On the other hand, kernels work as a medium between the operating system and the hardware within any system (Høiland-Jørgensen et al., 2018). In this case, the Linux kernel was chosen due to its performance, privacy, and flexibility to develop new applications (Passos et al., 2018).

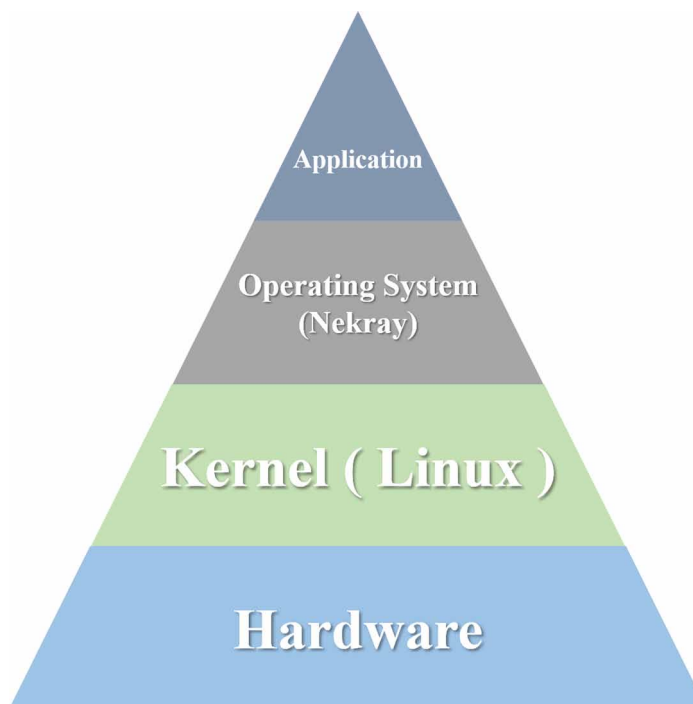
x64 Architecture

The x64 architecture of Nekray allows it to run in a system that can handle process 64-bits simultaneously. The following reasons were considered while choosing a 64-bit architecture for the OS:

- The OS was specifically developed for a 64-bit processor that allows extra address space and using the OS the processor can address significantly more memory than a 32-bit processor. This results in faster performance and a dramatic increase in the overall operation of the system.
- The x64 architecture allows the OS to handle more than 4-GB memory space. Therefore, there are no restrictions for the user of the OS regarding the memory space.
- The use of Kernel Patch Protection (KPP) can reduce the risk of kernel patching from harmful software.

Data Manipulation

After installation, Nekray can communicate with the kiosk camera to take its decisions. Along with the camera, other sensors can be added to the kiosk and the OS can easily read them. Currently, the OS supports the following sensors without any sort of additional programming:

Figure 1. OS development workflow

- Fingerprint Sensor
- Barometric Pressure Sensor
- Thermal Sensor
- Gas Sensor
- Humidity and Rain Detection
- Accelerometer Module
- Vibration switch module
- NFC
- Moisture Sensor
- Motion Sensor
- ultrasonic sensor
- RFID card reader
- GPS Module
- Gyroscope
- Compass
- Bluetooth/WIFI/GSM/Infrared diodes
- Heartbeat / Pulse Sensor
- Photoresistors

Hardware Requirements

Nekray was developed in a way that ensures easier installation and the required hardware to operate the kiosk is of quite a lower specification compared to most other operating systems (Sabri et al., 2017). The minimum requirements of the OS have been shown in Table 1.

Table 1. Hardware Requirements of the Nekray OS

Hardware	Minimum Requirements
CPU	1 GHz
RAM	1 GB
Disk Space	35 GB

Bootng Procedure

Figure 2 illustrates the steps followed by the OS in order to execute an application. Initially, the kernel compiles and checks the necessary hardware. Afterward, it runs the necessary codes to compile the OS. Finally, the applications can be executed from the home screen.

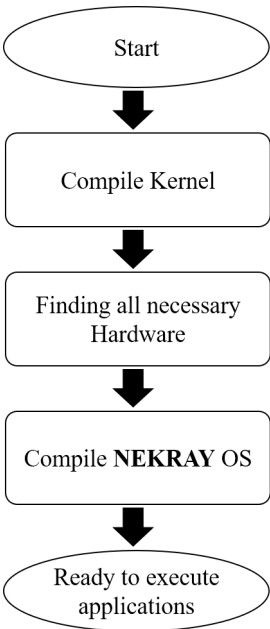
ACCESS MODE

The Nekray OS offers two types of access modes for the users and admins. These access modes are ‘User Mode’ and ‘Developer Mode’. Only authorized personals can access the ‘Developer Mode’, while the ‘User Mode’ can be used by everyone in general. The access mode of Nekray OS preserves the dynamicity and user-friendly features at the same time that hardly any other OS for Kiosks found in the market.

User Mode

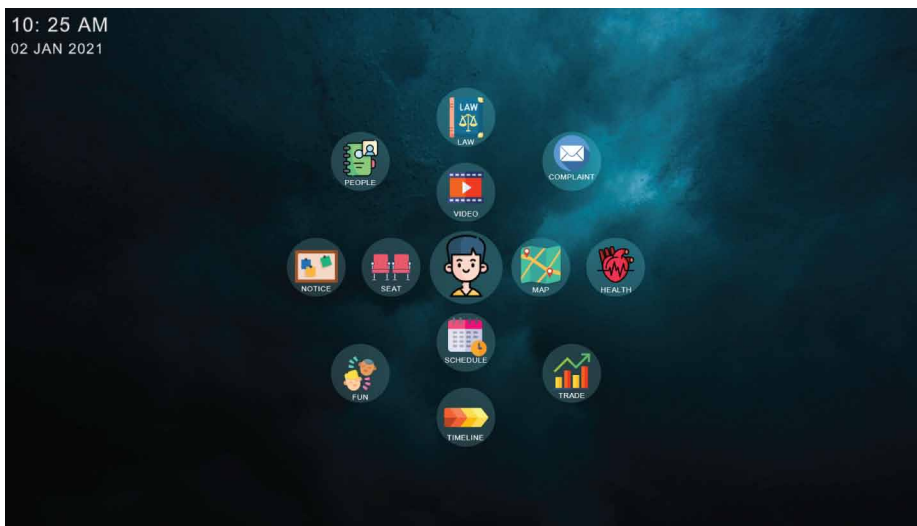
The user mode of the Nekray OS is the main outlook of the system developed for its users which is shown in Figure 3. Whenever the general users come in front of the kiosk device, this mode gets activated and performs different tasks for the users. The users can not make any sort of changes to the OS in this mode. Depending upon user category (i.e. Guest Users, Organization’s Users, Authorized

Figure 2. Application execution



Users), the user mode avails the range of features for its users. In addition, depending upon the age and gender of the user, the avatar at the center of the main menu will change and resemble accordingly. This detection is carried out by the image processing of the system. The avatar will be replaced by a pre-defined user's profile picture provided in the database of the system for authorized users. On the other hand, only avatars will be shown for guest users or the users whose data is not pre-defined in the database. The Nekray OS offers one of the most attractive user interfaces for its users that proves the system to be very user-friendly, user-interactive, and easy to use without any prior knowledge of its use.

Figure 3. Main Menu in User Mode



Developer Mode

The Nekray OS offers some of the most dynamic features in modern time for Kiosk devices. The developer mode of the Nekray OS is built for the admins or the administrative personals of an organization to modify the OS as per their needs and provides them the freedom to make changes in certain features of the system. The developer mode is basically featured in the Linux Kernal environment which is more like the existing Linux distribution found in the market. Therefore, the admins could easily make certain changes in the features as per requirement. The Developer Mode can be accessed by the admins through a specific shortcut key. For Nekray OS, this shortcut key was set in default as 'Ctrl+Alt+S'. After pressing this specified short key, a panel of username and password will appear on the screen where the admins need to provide the pre-specified user name and password to access the developer mode. As kiosk is a device mostly used on a touchscreen display, general users cannot access the developer mode through the touchscreen keypad. A keyboard needs to be plugged into the hardware device of the kiosk processor unit manually so that the specific shortcut key can be pressed and alter the access mode to access the developer mode from the user mode. The developer mode enables the admins to access the backend structure of the Nekray OS and allows them to make any necessary changes to the system. Through the developer mode, the admins can also make changes on different features such as 3D Map, 3D Seat-plan, Events, UI, etc.

CUSTOM FEATURES DEVELOPMENT

The developed Nekray OS is compatible to run different applications based on the requirement of any kiosks. In this case, different programs were developed for an information kiosk that can run on the OS architecture. The system was implemented These programs/features were developed using several frameworks (e.g. OpenCV, Electron, MySQL, etc.) and programming languages (e.g. Python, JavaScript, C#, etc.) Even though there were several frameworks and programming languages used in the development of the system, all of them were perfectly merged to present the final product. The UI and presentation of the program were developed in such a way that makes the device more user-interactive and user-friendly.

The main processing unit in terms of hardware used at the primary stage was the Raspberry Pi (Freeborn, 2019). However, as the Nekray was developed based on Linux kernel, the system can be installed in any other model of processing unit or computer hardware (e.g. Nvidia Jetson, Asus Tinker Board S, Odroid, Libre Computer Board, Orange Pi, LattePanda, etc.). To run different types of frameworks and programming languages in the computer board (Raspberry Pi 4B), the environment needed to be set up initially by installing different frameworks and software in it. These frameworks include Electron, TensorFlow Lite, Darkflow, OpenCV, JavaScript, etc.

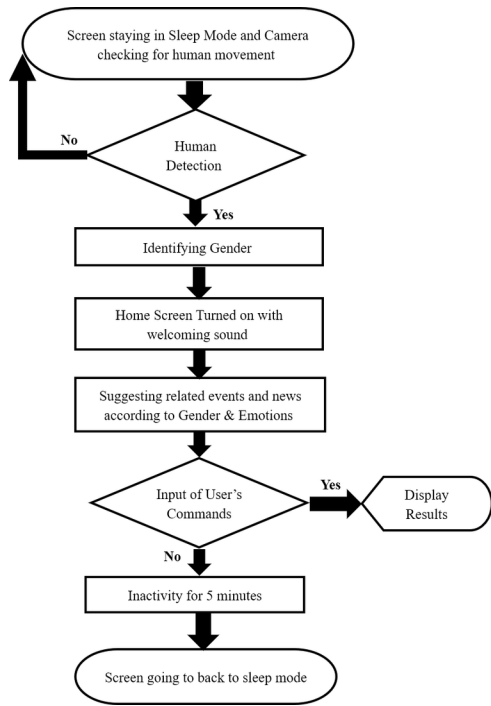
Image Processing

The Nekray carries out Image Processing features with high efficiency and fast speed. The features were developed using the Python framework of OpenCV, Tensorflow, YOLOv2, COCO Dataset, etc. Nekray carries out this image processing feature on the backend of the device in three parts: 1) Human Detection, 2) Facial Recognition, and 3) Age and Gender Prediction. These parts were developed using separate conditional programs and libraries. In this part of the image processing, the Nekray OS was developed with the Python framework of OpenCV. It allows real-time computer vision features with excellent accuracy. Basically, in this case, the OpenCV utilizes the SSD framework along with the ResNet as the core network. In our project, the library uses previously gathered human datasets. The system was proven to be highly efficient in detecting humans. The human detection algorithm was developed in a way that, when any human presence is in the range of the camera, the system recognizes the human through image processing whenever the detection level reaches at least 60%. The requirement of the detection algorithm was considered less to ensure consistency as the camera performance often varies due to low lighting conditions. After the detection, the system identifies the gender and emotions of the detected user. One of the hidden features of the system is by machine learning, the system would generate news and events notification based on a person's gender and current emotional state which can be determined by the captured image of the user's face. This image processing feature helps the system to become more power-saving and also user interactive at the same time. As the main power-consuming part of the device is the display screen itself, the system was designed to stay in sleep mode while there is no human presence nearby. The display screen lights up whenever any human is detected within the touching distance of the device. The screen then lights up with a welcoming sound that greets the user. If the user provides any command to the device for getting any information, the device performs that operation displays the desired result on the screen. When the user is done with the use of the device, the device will wait for 5 minutes to detect the further command by the user or detect any other human presence nearby. If there is no human presence detected, the screen will go back to sleep mode again and keep on scanning for any human detection. The flowchart showing the entire process has been shown in Figure 4. This development helps the device to consume less amount of power and also prevents power wastage by the device.

Human Detection

The operating system was developed in a way that after installation it can allow a system to detect human presence in front of it and lights up the display screen with a greeting sound. In this process,

Figure 4. Flowchart of image processing



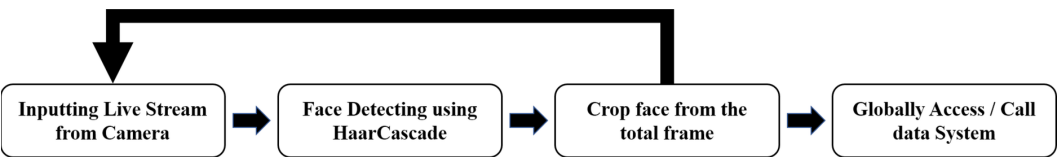
basic image processing was programmed to detect the human at a fair distance. For this, the object detection method was used to detect the human. The OpenCV and COCO Dataset in combination have a default dataset to recognize up to 93 different objects through image processing. Using this dataset, the human detections are carried out by the system. This method is shown in Figure 5.

Face Recognition

After constructing the initial environments for image recognition, Nekray was then trained up for Face Recognition. In some cases, Nekray was trained with a significant amount of sample to recognize some human subjects and identify their names and other information. For this feature, Computer Vision architecture was used for recognizing faces. Some of the highly efficient libraries were used to develop this program. Among these libraries, the following libraries can be highlighted to be recognized as mostly used ones:

- ‘face_recognition’ library from GitHub
- ‘Dlib C++ Library’
- ‘scikit-learn: Machine Learning in Python’

Figure 5. Human detection workflow



In real-time detection, Nekray uses computed 128-D face embeddings to detect a face from an image (Rosebrock, 2020). These images are collected by the system in real-time video data which then separates the images from each frame. Furthermore, Nekray was then trained with a Support Vector Machine (also known as the 'SVM') over the embeddings by 128-D. This way, the system can perform extraction on the 128-D featured vectors that are used for quantifying each face in a particular image where a combined implementation of Python and Torch was used for recognizing faces with the help of Deep Learning (Schroff et al., 2015). Using this model, Nekray was attained with greater efficiency on the purpose of representation and performing state-of-the-art face recognition by the use of only 128-D.

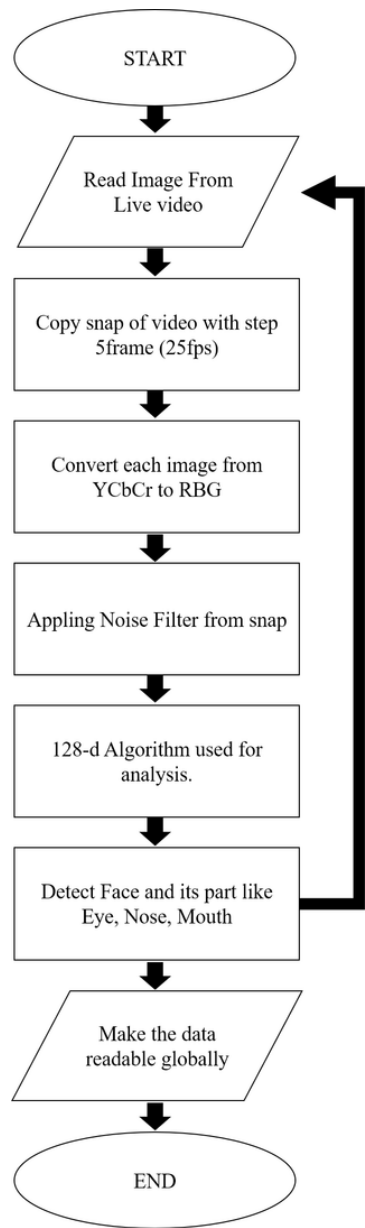
In practice, several steps were followed for image recognition and face detection. Firstly, the Nekray needed to be provided with an image or video as the input to run the process. Once the input was given to the face recognition pipeline, the Nekray starts to analyze it and then applies the process of face detection in order to detect the location of a face in that particular image. In the case of video, the system splits apart images from the video frame by frame. To make the system gain more efficiency, Nekray can compute the 'Facial Landmarks' as an option that enables it to pre-process and align the face (Johnston & de Chazal, 2018). This particular process of face alignment demonstrates the increase of accuracy of the face recognition in some of the pipelines. After applying the face alignment which stays as an option, as well as the cropping; the image of the face, which is given as the input is then passed through Nekray's Deep Neural Network. With the help of FaceNet, Nekray can directly learn mapping from the images of a face to a compact Euclidean Space. Here, the distances between coordinates directly correspond to a measurement of the similarity of the face.

After setting up the program as per the requirements for detecting a face and identifying it with higher accuracy, Nekray was developed even further so that it can be trained with some particular dataset and recognize some users by recognizing their faces. In order to train a model of face recognition with deep learning, each input batch of data is identified by processing them like three types of images. At this stage, the system extracts the 128-D embeddings for each face. After that, Nekray was needed to be trained with a 'standard' model for machine learning (such as SVM, K-NN, Random Forest, etc.) over the embeddings layer. We used some datasets and human subjects to train the system. The overall workflow has been shown in Figure 6.

The three types of images used for training the Nekray system could be defined with the following mentioned points:

- **The Anchor:** It is the edges or anchors of the current face given to the system which has an identity. The Nekray system generates the anchor box around the face with a certain height and width. This box is then used for defining the particular face image's aspect ratio, scalings, and other geometrical attributes of the image. It then identifies a subject with any identity names given to the subject's database in the system.
- **The Positive Image:** The positive image focuses on the identified image (e.g. Face) of the subject image and adjusts some of the attributes of the image (e.g. Contrast, Saturation, Sharpening, Gradient), and darkens the focused area of the subject's image to identify the face. In other words, it turns the focused area by transforming it into a darkened black area and then recognizes the face according to the dataset.
- **The Negative Image:** The Negative Image works similarly to the Positive Image. It also focuses on the identified image (e.g. Face) of the given subject's image and adjusts some of the image attributes (e.g. Contrast, Saturation, Sharpening, Gradient), and brightens the focused area of the focused area. In other words, it turns the focused area by transforming it into a bright white area and then recognizes the face according to the dataset.

Figure 6. Face recognition workflow



The overall accuracy of Nekray in terms of image processing depends on the amount of data it is trained with. With a sufficient amount of data, the Nekray system performs with higher efficiency and shows better results in detecting the users.

Age Detection

One of the advanced features added to Nekray’s Image Processing is the ‘Age Detection’. The Nekray system can identify and predict a person’s age from the image or live video feed as shown in Figure

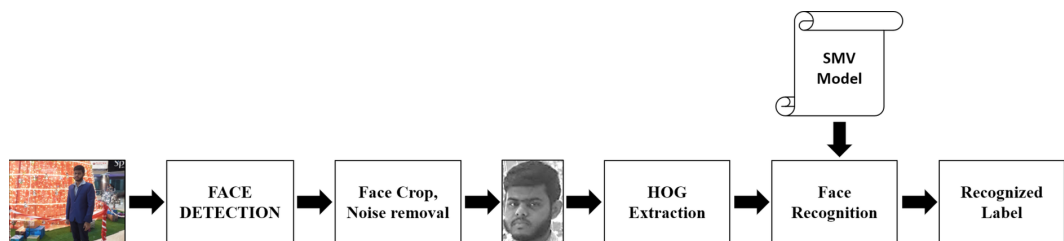
7. This is a process where the system automatically detects the age of a person by only analyzing the photo of their face. This feature is carried out using Computer Vision.

The Age Detection feature was developed in two-stages:

Stage 1: In the first stage of the Age Detection process, the system detects faces from the image/video stream which is given as the input to the system.

Stage 2: On the second stage of the Age Detection process, the system extracts the face Region of Interest (ROI) and applies the Age Detector algorithm to find the age of the person.

Figure 7. Age detection



For performing tasks at Stage 1, any face detector which can generate bounding boxes for faces in a photo is needed for the process. There are numerous face detector algorithms available capable of such procedure; for example, Haar Cascades, HOG + Linear SVM, Single Shot Detectors (SSDs), etc.

Haar Cascades performs at quite a significantly faster speed and has the capacity of running on embedded devices in real-time (Mantoro & Ayu, 2018). But the drawback with the Haar Cascade is that it produces less accurate results and it shows the characteristics of being prone at a significant rate to false-positive detections. HOG + Linear SVM models perform more accurately than the Haar Cascades, but they process their tasks at a relatively slower speed. Besides, they are not as tolerant of occlusion, which may produce results that don't have the whole face visible. It also couldn't produce accurate results due to the viewpoint changes (i.e. Views from different angles on the face producing different views on the facial images).

Face Detectors that are developed based on Deep Learning are significantly more robust than others and more likely to perform with higher accuracy. But it is required for them to have more computational resources in comparison with both Haar Cascade and HOG + Linear SVMs. Considering these factors, and also by judging the speed and accuracy of the face detectors; the Nekray system was developed using the SSD (Single Shot Detector) model. This is a Deep Learning-based model that requires a lot of computational resources but provides significantly higher accuracy in face detection. The model splits apart images from a video feed by processing it frame by frame. Therefore, in each frame, it takes in a single image and then runs the algorithm process on it to detect the face. This method proved to be quite efficient in face detection which ultimately helps Stage 2 for detecting the age of a person.

Once the Face Detector produces the coordinates of the bounding box which is generated on the face in the image/video stream; the system then can advance to the Second Stage of the process, which is the identification of the age of the person. From the given bounding box on an image, the (X, Y) – Coordinates of the generated box on the face are used initially to extract the face ROI. During this procedure, the algorithm ignores the rest of the images/frames. By this procedure, the system enables the Age Detector to make focus specifically on the person's face and not anywhere else in the

image nor on any other irrelevant noises on the particular image. After that, the face ROI is passed through the model, carrying out a genuine prediction on the person. There are several algorithms for Age Detection available. But the Deep Learning-based Age Detectors stand out to be more popular and efficient than any others. For the Nekray system, a Deep Learning-based age detector model was chosen [8]. Here, a simple AlexNet-like architecture was used, that goes through the training procedure to learn & detect a total number of 8 age brackets. These age brackets are: (1) 0-2, (2) 4-6, (3) 8-12, (4) 15-20, (5) 25-32, (6) 38-43, (7) 48-53, and (8) 60-100. These age brackets have proved to be noncontiguous and it was carried out intentionally, as the 'Adience Dataset' is used for training the model, defining the specified ranges of age as such. For the Nekray system, a pre-trained age detector model was used.

Embedded Multiple Sensors

For collecting and providing real-time environmental and biological data, the Nekray system was developed to control multiple sensors at a time and effectively produce the results processing both analog signals as well as digital signals as shown in Figure 8. To control these multiple sensors, a signal converter was designed for the system that converts analog signal to digital signal and then fed to the system CPU for further processing. This developed converter for Nekray OS was named as '**Nekray Module**'. This Nekray Module can be replaced with any other microcontroller (e. g. Arduino Uno, Arduino Mega, etc.) but that would require the microcontroller to be coded accordingly with a necessary platform (e.g. Arduino CC). The Nekray Module is connected to multiple sensors that collect different types of biological and environmental data. The collected data would then be transferred to the Nekray Module as an analog signal or digital signal. The module would convert all the received signals into a digital signal and send it to the system CPU through serial communication. For the Nekray OS, relevant code is developed for specified sensors that are used with the kiosk device. Different codes specified for individual sensors are merged into a unified code that can control all the sensors simultaneously. The number of sensors that could be used with the Nekray OS is not fixed. Instead, this feature was also kept as dynamic where required sensors could be added to the system at any given time with necessary modification in the code and circuit diagrams.

3D Map Development

3D Map Development is one of the most exclusive features brought into the Nekray system. This enables the users to navigate through the map to know where the desired destination is located and

Figure 8. Workflow of embedded multiple sensors

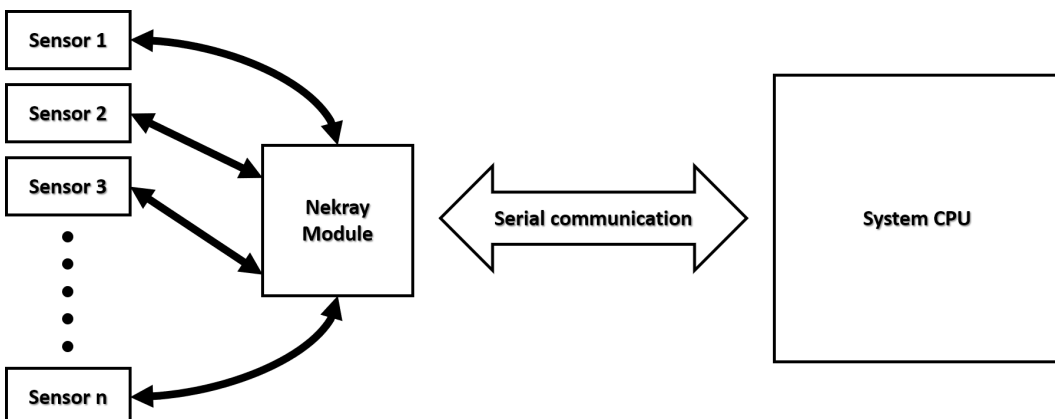
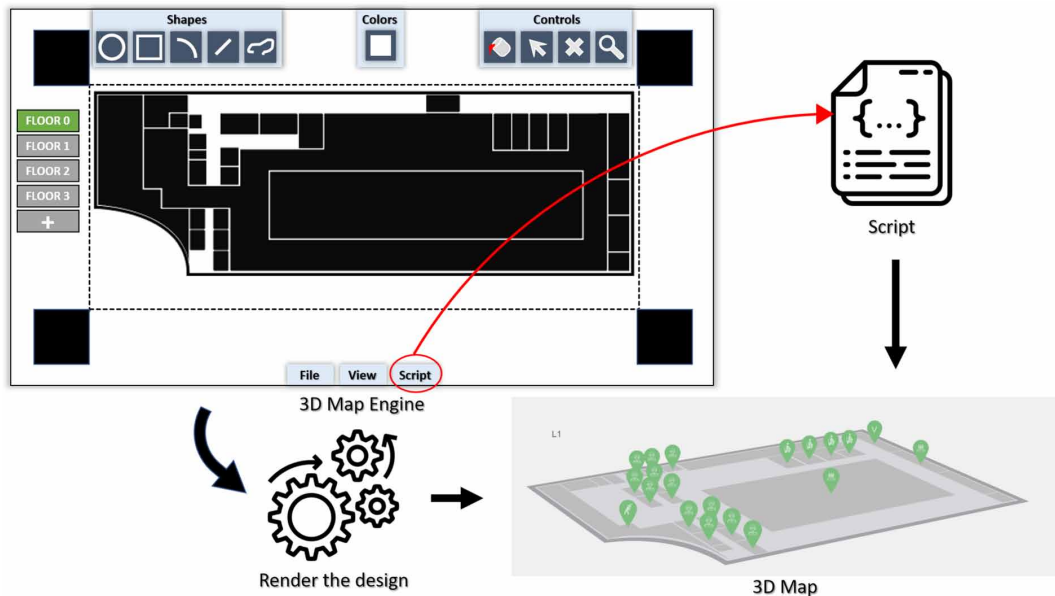


Figure 9. Development of 3D map in Nekray



how to get there. The 3D Map was mainly developed using JavaScript programming. The 3D Map was developed in three stages. These stages are explained in the following sections.

Structural Architecture Design

At this stage, the model organizational building's whole architecture is designed with the 3D Map Engine, which is available in the developer mode of Nekray by default. The structural design gets created according to the actual measurements of the floors and spaces of the model building. The final developed architectural design is later used for creating the 3D Map using the measurement of the floors by putting them through the script of the 3D Map.

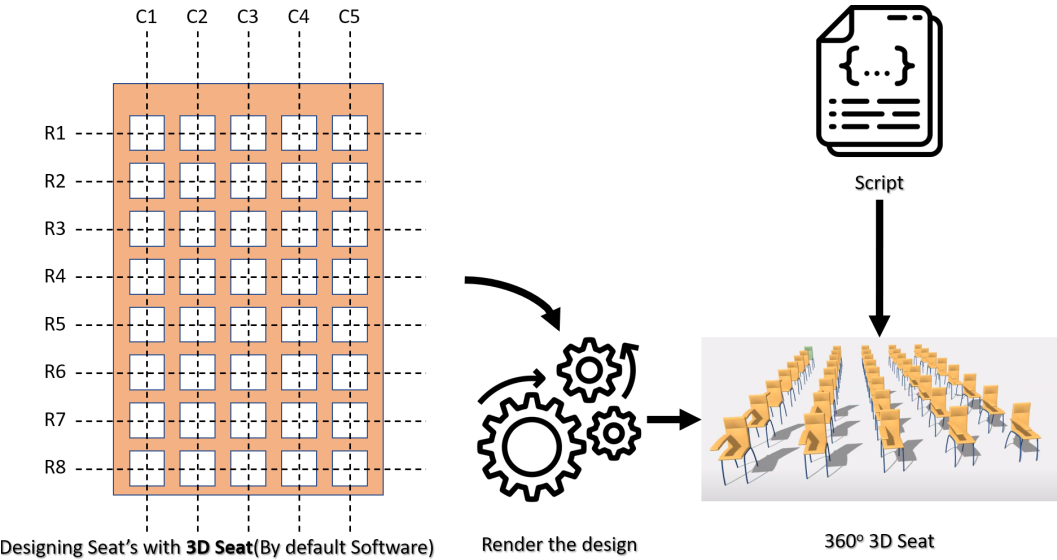
Database of the 3D Map

At this particular stage, the data of the model building was collected precisely. These data include the following: Room Numbers, Person's Name, Room Names, Email addresses of the person, person's designation in the organization, etc. These data are provided according to the data of the targeted organization or institution and used for developing the database for the 3D Map. Later, these data are used for developing the final version of the 3D Map that includes all of these data. Users while using the 3D Map feature can gain information regarding any location on the map and also can understand how to get there.

Programming and Developing the 3D Map

This is the final stage of the feature where all data from the first two stages came together and was used to develop the final program. Basically, all the data gained from the first two stages were used in the coding of the program. These coding ultimately generated the final 3D Map. The 3D Map could be navigated in accordance with the needs of the user. One of the exclusive features Nekray provides is the dynamicity of the 3D map. According to the data of any organization, the 3D map can be designed and modified accordingly quite simply at any time. The overall procedure of the 3D map development has been shown in Figure 9.

Figure 10. Development of 3D seat-plan in Nekray



Development of 3D Seat Plan

The 3D Seat Plan feature was also developed based on the same work or concept as it was used in the 3D Map as shown in Figure 10. In this program, a default room model was designed with a given number of seats. The structure of the room and seats, in this case, are also dynamic which can be changed according to any organization or institution (University, Airport, Bus Station, Rail Station, Office, Seminar Hall, etc.). In this feature, the seats were first designed with a 3D Seat Engine by 2D graphics. Then, a single-seat or chair design was copied to make any given number of seats in the room. A positional matrix was developed in the program to identify each chair's location in the classroom. A backend database was also programmed which has all user's data in it. Whenever any particular user's data is searched in the system, that user's data will be fetched by the program and locate the assigned seat of the user in the room.

RESULTS ANALYSIS

This section evaluates the performance of the developed operating system by calculating the accuracy of the used algorithms and comparing the performance of the developed OS with existing operating systems. To test the accuracy of the artificial intelligence used by the OS, accuracy, performance, recall and F1 score have been measured. In addition to that, to justify the development of a new operating system for kiosks, we focused on the resource utilization of the existing operating systems and compared them with the Nekray OS.

Table 2. Dataset for human detection

Task: Human detection	Actual class	Predicted class		
			Class = Yes	Class = No
		Class = Yes	True positive (TP) = 47	False negative (TN) = 8
		Class = No	False positive (FP) = 4	True negative (TN) = 63

Table 3. Dataset for gender detection

Task: Gender Detection	Actual class	Predicted class		
			Class = Yes	Class = No
		Class = Yes	True positive (TP) = 36	False negative (FN) = 11
		Class = No	False positive (FP) = 16	True negative (TN) = 59

AI Performance

Artificial intelligence is one of the core features of Nekray OS to detect humans and to present them with the necessary information easily and in a more user-friendly manner. As mentioned in the earlier step, Nekray was trained with a Support Vector Machine (SVM) to recognize human faces with the help of Deep Learning. To test its performance total of 122 datasets have been used as shown in Table 2. The datasets were used to test the ability of the algorithms to predict the actual subject correctly.

Using the dataset from Table 2, the following results can be obtained:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} = 0.9016$$

$$\text{Precision} = \frac{TP}{TP+FP} = 0.9215$$

$$\text{Recall} = \frac{TP}{TP+FN} = 0.8545$$

$$\text{F1 Score} = \frac{2 * (\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} = 0.8867$$

These results show that the OS can accurately handle the task of human detection in most cases.

Gender Prediction

Nekray OS uses the camera data to detect the user's gender and emotion so that the system can profile the user. This way, the system can suggest news and events related to the user. In this stage, the accuracy of this task will be evaluated. Following the same strategy as for human detection, similar datasets were used to test the ability of the algorithms to predict the gender of the subject as shown in Table 3.

The following results can be obtained from the retrieved data:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} = 0.7786$$

$$\text{Precision} = \frac{TP}{TP+FP} = 0.6923$$

$$\text{Recall} = \frac{TP}{TP+FN} = 0.7659$$

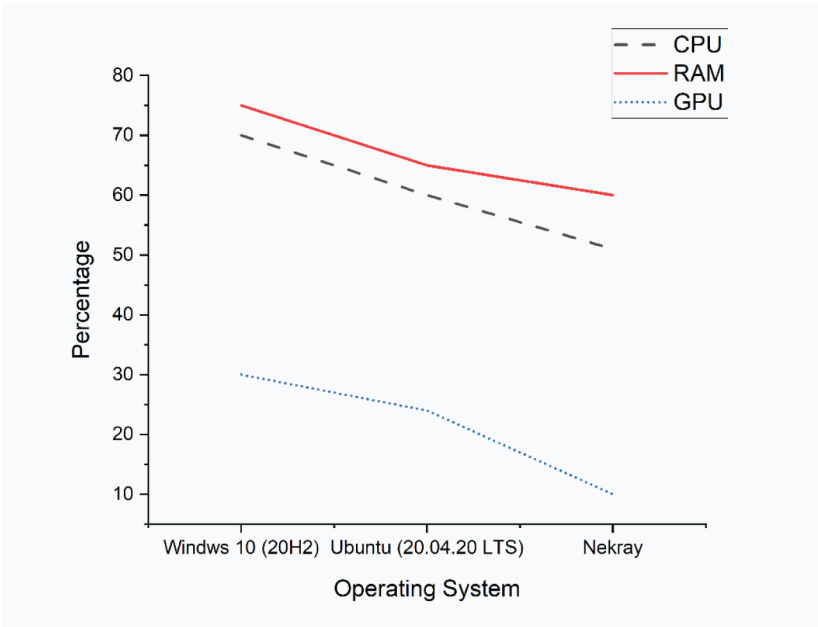
$$\text{F1 Score} = \frac{2 * (\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} = 0.7272$$

Based on the obtained results, it can be said that the accuracy of the gender detection by the Nekray OS is pretty high, although the accuracy is lower than the human detection.

Hardware Resource Utilization

In Figure 11, the resource utilization of different hardware using three different operating systems (Windows 10, Linux, Nekray) during the working of the information kiosk has been shown. These data are obtained while the kiosk was running with hardware consisting of a 1.5GHz quad-core ARM Cortex A72 chipset, a 4GB ram, and a 164 MB video memory. A similar type of evaluation was also performed in earlier research (Boras et al., 2020). To get accurate data, the same tasks were performed individually by each of the operating systems. The result shows that Nekray OS uses less CPU, RAM, and GPU memory than the rest of them while working on a kiosk. This can be pretty hand in the case of a kiosk as kiosks are built to be performed continuously without any interruption.

Figure 11. Hardware utilization by different OS



Survey Findings

A survey was conducted to find the user experience on the developed Nekray OS. More than 1500 people participated in this survey after having real-time experience using the developed kiosk device with Nekray OS. The users were mostly students of a university (American International University-Bangladesh) and visiting guests to the university. A major portion of the people gave a positive (91%) response that the kiosk was easier to use than the existing kiosks as shown in Figure 12. This was the

Figure 12. Survey findings on the usability of Nekray OS

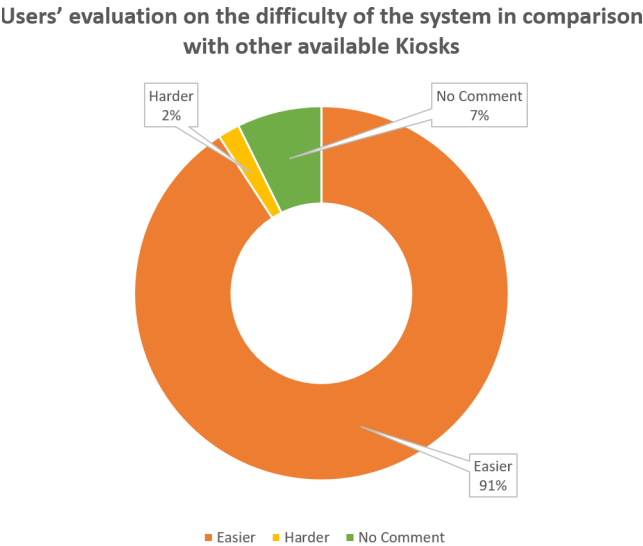
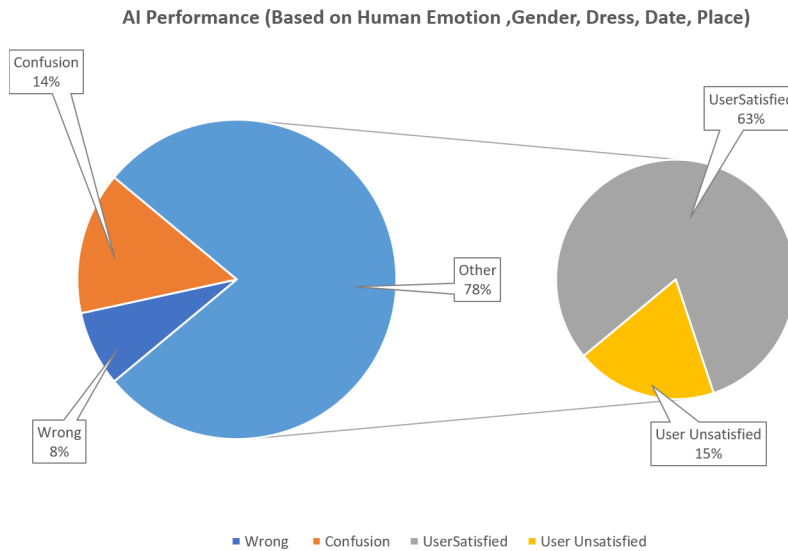


Figure 13. Survey findings on the AI performance of Nekray OS



primary target while developing the OS; to build a user-friendly OS. 63% percent of the users were satisfied with the accuracy of the built-in AI feature of the kiosk to detect their face and emotion as shown in Figure 13.

CONCLUSION

This paper presents the development of an operating system based on the Linux kernel for kiosk devices. After investigating the current state of operating the systems, a broad description of different aspects of the developed OS has been provided. Different features that can run on the OS have been also developed based on the requirements of information kiosks. As information kiosks require obtaining fast and accurate data for its user, the OS was developed using the Linux kernel, which is widely popular for its performance and usability. The key motivation behind developing this OS was to reduce hardware resource usage during the operation of information kiosks, reducing the installation costs of such kiosk devices, and making an attractive UI for a better user experience. The objective was successfully achieved as the OS significantly used fewer hardware resources than existing operating systems while performing daily operations. The OS was also successful to identify human movement and gender with the use of different algorithms like SVM and achieved an accuracy of 90% and 78% consecutively. The developed OS differs from the existing kiosk operating systems based on its user-interface, hardware utilization, and machine learning applications. The future scope for this research is huge the image processing capability of the OS can be improved and the impact of AI to reduce the power consumption of electronic devices can be also investigated in the future.

REFERENCES

- Afzal, B., Umair, M., Asadullah Shah, G., & Ahmed, E. (2019). Enabling IoT platforms for social IoT applications: Vision, feature mapping, and challenges. *Future Generation Computer Systems*, 92, 718–731. doi:10.1016/j.future.2017.12.002
- Afzali, M., Ahmadi, M., & Mahmoudvand, Z. (2017). Data Requirements and the Basis for Designing Health Information Kiosks. *Acta Informatica Medica*, 25(3), 198. PMID:29114115
- Boras, M., Balen, J., & Vdovjak, K. (2020). Performance Evaluation of Linux Operating Systems. *2020 International Conference on Smart Systems and Technologies (SST)*, 115–120. doi:<ALIGNMENT.qj></ALIGNMENT>10.1109/SST49455.2020.9264055
- Branzila, M., Sarmasanu, C., Ciudin, D., & Lacatusu, D. (2018). Sensors and transducers laboratory kiosk. *2018 International Conference and Exposition on Electrical And Power Engineering (EPE)*, 377–380.
- Choi, E., & Hong, J. (2019). Design and implementation of virtual machine control and streaming scheme using Linux kernel-based virtual machine hypercall for virtual mobile infrastructure. *Proceedings of the Conference on Research in Adaptive and Convergent Systems*, 57–60. doi:<ALIGNMENT.qj></ALIGNMENT>10.1145/3338840.3355690
- Choyon, M. M. S., Rahman, M., Kabir, M. M., & Mridha, M. F. (2020). IoT based Health Monitoring Automated Predictive System to Confront COVID-19. *2020 IEEE 17th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET)*, 189–193. https://doi.org/doi:10.1109/HONET50430.2020.9322811
- Ekşioğlu, M., Güler, H., Terzi, F., Yıldırım, H. S., & Yücel, B. (2018). *UXD for a Prototype Campus Information Kiosk*. Academic Press.
- Freeborn, T. J. (2019). Performance evaluation of raspberry Pi platform for bioimpedance analysis using least squares optimization. *Personal and Ubiquitous Computing*, 23(2), 279–285.
- Gerofi, B., Riesen, R., Takagi, M., Boku, T., Nakajima, K., Ishikawa, Y., & Wisniewski, R. W. (2018). Performance and scalability of lightweight multi-kernel based operating systems. *2018 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 116–125.
- Gladia, R. A., & Kavya, G. (2017). Design and Development of Non-Invasive Kiosk for Self-Care Health Management. *Global Journal of Pure and Applied Mathematics*, 13, 4787–4794.
- Graham, K., Fai, S., Dhanda, A., Smith, L., Tousant, K., Wang, E., & Weigert, A. (2018). The VR kiosk. In *Digital Cultural Heritage* (pp. 324–336). Springer.
- Han, J., Oh, Y., & Ham, S. (2019). Influence of Ordering Kiosk Nutrition Information Transparency and Information Quality on the Customer Behavioral Intention in Fast Food Restaurants. *Journal of the Korean Dietetic Association*, 25(3), 165–177.
- Høiland-Jørgensen, T., Brouer, J. D., Borkmann, D., Fastabend, J., Herbert, T., Ahern, D., & Miller, D. (2018). The express data path: Fast programmable packet processing in the operating system kernel. *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies*, 54–66.
- Johnston, B., & de Chazal, P. (2018). A review of image-based automatic facial landmark identification techniques. *EURASIP Journal on Image and Video Processing*, 2018(1), 86.
- Joshi, A., Arora, M., & Malhotra, B. (2017). Usability Evaluation of a Portable Health Information Kiosk Using a SMAARTTM Intervention Framework. *Global Journal of Health Science*, 9(8), 153. doi:10.5539/gjhs.v9n8p153
- Kum, S. W., Moon, J., & Lim, T. (2017). Design of fog computing based IoT application architecture. *2017 IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*, 88–89. doi:<ALIGNMENT.qj></ALIGNMENT>10.1109/ICCE-Berlin.2017.8210598
- Liu, K., Yang, M., Ling, Z., Yan, H., Zhang, Y., Fu, X., & Zhao, W. (2020). On Manually Reverse Engineering Communication Protocols of Linux Based IoT Systems. *IEEE Internet of Things Journal*, 1–1. doi:10.1109/JIOT.2020.3036232

- Passos, L., Queiroz, R., Mukelabai, M., Berger, T., Apel, S., Czarnecki, K., & Padilla, J. (2018). A study of feature scattering in the linux kernel. *IEEE Transactions on Software Engineering*.
- Payne, B. R., & Abegaz, T. T. (2018). Securing the Internet of Things: Best Practices for Deploying IoT Devices. In K. Daimi (Ed.), *Computer and Network Security Essentials* (pp. 493–506). Springer International Publishing. doi:10.1007/978-3-319-58424-9_28
- Rajendran, P. S. (2018). Virtual information kiosk using augmented reality for easy shopping. *International Journal of Pure and Applied Mathematics*, 118(20), 985–994.
- Rosebrock, A. (2020). *Raspberry Pi and Movidius NCS Face Recognition—PyImageSearch*. Pyimagesearch. <https://www.pyimagesearch.com/2020/01/06/raspberry-pi-and-movidius-ncs-face-recognition/>
- Sabri, C., Kriaa, L., & Azzouz, S. L. (2017). Comparison of IoT Constrained Devices Operating Systems: A Survey. *2017 IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*, 369–375. doi:<ALIGNMENT.qj></ALIGNMENT>10.1109/AICCSA.2017.187
- Sad, A. M. H., Choyon, M. M. S., Rhydwan, A. H. M., & Hossain, C. A. (2020). An Interactive Low-Cost Smart Assistant System: Information Kiosk as Plug Play Device. *2020 27th Conference of Open Innovations Association (FRUCT)*, 193–199. doi:<ALIGNMENT.qj></ALIGNMENT>10.23919/FRUCT49677.2020.9211057
- Schroff, F., Kalenichenko, D., & Philbin, J. (2015). *FaceNet: A Unified Embedding for Face Recognition and Clustering*. https://openaccess.thecvf.com/content_cvpr_2015/html/Schroff_FaceNet_A_Unified_2015_CVPR_paper.html
- Serra, G., Ara, G., Fara, P., & Cucinotta, T. (2020). An Architecture for Declarative Real-Time Scheduling on Linux. *2020 IEEE 23rd International Symposium on Real-Time Distributed Computing (ISORC)*, 20–28. doi:<ALIGNMENT.qj></ALIGNMENT>10.1109/ISORC49007.2020.00013
- Swain, M., Singh, R., & Hashmi, Md. F., & Gehlot, A. (2020). Performance Analysis of Various Embedded Linux Firmwares for ARM Architecture Based IoT Devices. In G. Singh Tomar, N. S. Chaudhari, J. L. V. Barbosa, & M. K. Aghwariya (Eds.), *International Conference on Intelligent Computing and Smart Communication 2019* (pp. 1451–1460). Springer. doi:<ALIGNMENT.qj></ALIGNMENT>10.1007/978-981-15-0633-8_143
- Vishnubhatla, A. (2020). Linux Server Based Automatic Online Ticketing Kiosk. In A. P. Pandian, R. Palanisamy, & K. Ntalianis (Eds.), *Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2019)* (pp. 428–439). Springer International Publishing. doi:<ALIGNMENT.qj></ALIGNMENT>10.1007/978-3-030-43192-1_49
- Zhang, S., Meng, X., Wang, L., & Liu, G. (2017). Research on Linux Kernel Version Diversity for Precise Memory Analysis. In B. Zou, M. Li, H. Wang, X. Song, W. Xie, & Z. Lu (Eds.), *Data Science* (pp. 373–385). Springer. doi:<ALIGNMENT.qj></ALIGNMENT>10.1007/978-981-10-6385-5_32

A. S. M. Mehedi Hasan Sad completed his Bachelor of Science in Electrical & Electronic Engineering from American International University-Bangladesh in 2020. His research interests are Circuit Designing, Computer Vision, Smart System, IoT, Embedded System, Polymer-based Devices, Electronics, Automation, and Robotics. Currently he works as a Circuit and System Design Engineer in a semiconductor industry.

Md Mashrur Sakib Choyon completed his Bachelor of Science in Electrical & Electronic Engineering from American International University-Bangladesh in 2020. He is a student member of IEEE. His research interests are: Smart System, IoT, Circuit Designing, Embedded System, Polymer-based Devices, Electronics, Automation, and Robotics.

Abu Hasnat Md Rhydwan completed his Bachelor of Science in Electrical & Electronic Engineering from American International University-Bangladesh in 2020. His research interests are: Smart Grid, Renewable Energy, and Power Electronics.

Kawshik Shikder is currently working as an Assistant Professor in Dept. of EEE, Faculty of Engineering in American International University-Bangladesh (AIUB). He received his Master degree in Electrical & Electronic Engineering (M.Sc in EEE) with "Summa-Cum-Laude" gold medal award from AIUB in 2015. He achieved his Bachelor degree in Electrical & Electronic Engineering (B.Sc in EEE) from the same university in 2013. Mr. Shikder has so far published 4 journal, 13 conference papers and a few more have been submitted for review in different journals and conferences. His main research focus is on semiconductor materials, materials for wearable devices, materials for highspeed/high bandwidth devices, Nanofabrication, Nanoelectronics and Opto electronic Devices.

Chowdhury Akram Hossain received his B.Sc. degree in Electrical and Electronics Engineering (EEE) and M.Engg. degree (with the honour of Summa Cum Laude) in Telecommunication from the American International University-Bangladesh (AIUB), Dhaka, in 2008 and 2010 respectively. He is currently pursuing a Ph.D. degree in Telemedicine at the Universiti Sultan Zainal Abidin, Malaysia. In 2010, he joined as a Special Assistant under the Office of Student Affairs, and a Senior Lecturer for the Electrical and Electronics Engineering Department, under the Faculty of Engineering at AIUB, where he is currently serving as a Senior Assistant Professor. His research interests are in the field of power engineering, wireless communication, renewable energy, smart grid, Adhoc Network, etc. Besides being a Senior Member of IEEE, he has served as VOLT mentor in past years and currently he is a member of IEEE MGA Training committee.

International Journal of Embedded and Real-Time Communication Systems

Volume 12 • Issue 3 • July-September 2021 • ISSN: 1947-3176 • eISSN: 1947-3184

MISSION

The mission of the **International Journal of Embedded and Real-Time Communication Systems (IJERTCS)** is to disseminate recent advancements and innovations in this interdisciplinary research area for field researchers, practitioners, scientists, academicians, students, and IT professionals. IJERTCS focuses on overcoming challenges involved in the rapid development of embedded communication systems towards feature-rich multimedia computers.

SUBSCRIPTION INFORMATION

The International Journal of Embedded and Real-Time Communication Systems (IJERTCS) is available in print and electronic formats and offers individual or institution-level pricing. Full subscription information can be found at www.igi-global.com/IJERTCS.

IJERTCS is also included in IGI Global's InfoSci-Journals Database which contains all of IGI Global's peer-reviewed journals and offers unlimited simultaneous access, full-text PDF and XML viewing, with no DRM. Subscriptions to the InfoSci-Journals Database are available for institutions. For more information, please visit www.igi-global.com/infosci-journals or contact E-Resources at eresources@igi-global.com.

CORRESPONDENCE AND QUESTIONS

EDITORIAL

Sergey Balandin, Editor-in-Chief • IJERTCS@igi-global.com

SUBSCRIBER INFO

IGI Global • Customer Service

701 East Chocolate Avenue • Hershey PA 17033-1240, USA

Telephone: 717/533-8845 x100 • **E-Mail:** cust@igi-global.com

The International Journal of Embedded and Real-Time Communication Systems is indexed or listed in the following.

ACM Digital Library; Bacon's Media Directory; Cabell's Directories; DBLP; Google Scholar; INSPEC; JournalTOCs; Library & Information Science Abstracts (LISA); MediaFinder; SCOPUS; The Standard Periodical Directory; UGC-CARE List (India); Ulrich's Periodicals Directory; Web of Science