Received: November 2, 2024. Revised: December 16, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

RPSA-YOLOv10: Relative Partial Self-Attention for Object Recognition in Smart Glasses Based on Contextual Adaptation

Aradea[1]* **Irfan Darmawan[2]** **Rianto[1]** **Ghatan Fauzi Nugraha[1]**

[1]*Department of Informatics, Faculty of Engineering University of Siliwangi, Indonesia*
[2]*Department of Information System, Telkom University, Bandung, Indonesia*
* Corresponding author's Email: aradea@unsil.ac.id

Abstract: Uncertainty triggers everyone involved in activities in this world to adapt well, including visually impaired people. Therefore, this paper proposed a smart glasses model based on the Self-Adaptive Cyber-Physical System to help visually impaired people live their days. This model was equipped with object recognition capabilities as an extension of YOLOv10m and Relative Positional Encoding (RPE) where it was placed in the attention module and subsequently combined with Partial Self-Attention (PSA) to create a better understanding of spatial features compared to the attention module previously. Therefore, we introduced a new variant called Relative Partial Self-Attention (RPSA) YOLOv10. Our model indicated adaptability based on contextual knowledge, such as calculating object distances and the ability to work at low light intensity. Additionally, our model also operates through voice commands and voice notifications. The evaluative results of the model trained with the MS COCO dataset signified mAP50, mAP75, and mAP50-95 values of 67.3%, 54.5%, and 50.2% respectively with an inference speed of 8.4 ms/image. These results demonstrated better performance compared to other versions of the YOLO model, notably in evaluations using small datasets with an increase in mAP50-95 of 30.8% compared to the YOLOv10 model. In addition, our designed adaptive system can handle the problem of estimating object distances with an average error of 15.95%. Further, it can work on light intensity problems with a stable increase in average brightness reaching 95.65.

Keywords: Contextual knowledge, RPSA-YOLOv10, Self-adaptive cyber-physical system, Smart glasses.

1. Introduction

The rapid development of technology requires everyone to be able to adapt, including visually-impaired people. These demands make it quite challenging for visually impaired people to even get through their days [1]. World Health Organization (WHO) reported that around 2.2 billion worldwide experience visual impairment, including blindness [2]. This proves that many people suffering from eye problems require a solution. Myriad investigative results offer assistant devices as a solution to this problem [3, 4], and can be divided into two types, namely wearable devices and non-wearable devices. Wearable devices are often selected by researchers because they provide better practical functions compared to non-wearable devices [4]. Another factor is that wearable devices are more flexible when combined with artificial intelligence (AI) [1]. Moreover, one type of assistant device (smart glasses) is the wearable device most widely developed by researchers [5, 6].

Generally speaking, the advancement of smart glasses is equipped with object detection/recognition features as replacement eyes for impaired people [5, 7]. This feature is considered the most effective in smart glasses architecture. However, most of them still apply old models, such as YOLOv3, Faster-RCNN, SSD MobileNet v2, ResNet, etc. [1, 6, 8]. Apart from that, other features can be added, such as voice feedback, speech recognition, and distance estimation to enhance the functionality of smart glasses. Unfortunately, the speech recognition and distance estimation features still receive less attention from researchers, notably the distance estimation feature [6]. In this feature, the calculation process is generally focused on the employment of sensor modules [9, 10]. This can reduce the practicality of smart glasses devices.

Based on this description, there are still gaps providing opportunities for our research. One of them is fostering object detection capabilities on smart glasses with YOLOv10 [11] and modifying the attention mechanism to boost model performance. This modification was performed by combining the Partial Self-Attention (PSA) module used in YOLOv10 with Relative Positional Encoding (RPE). The addition of RPE was undertaken so that the attention module embedded in YOLOv10 was able to extract spatial features better than before. As a result, this modification was expected to help improve the performance of the object detection model in conducting its task of recognizing various objects and obstacles for visually impaired people. Besides, we included adaptability by developing object distance estimates without sensors and adaptive light detection to overcome uncertainty caused by an extension of the Self-Adaptive Cyber Physical System (SACPS) rule [12]. Moreover, we also targeted the addition of voice feedback and speech recognition features to raise the functionality of the smart glasses.

This paper consists of several main parts, namely introduction (first part), related works explaining the latest studies related to smart glasses for the visually impaired people (second part), proposed method containing our proposed methods and novelties (third part), experiments undertaken to validate the results of our designed method and novelty (fourth section), and conclusion from the entire research (fifth section).

2. Related works

2.1 Based on deep learning

Currently, the employment of deep learning-based object detection is a predominant component of environmental recognition in smart glasses. For instance, Islam et al. [13] cultivated object detection with the SSDLite MobilNet v2 model and trained with the Microsoft Common Object in Context (MS COCO) dataset and hyperparameter optimization with particle swarm optimization (PSO). The model produced 88.89% accuracy with a real-time processing speed of 2.15 FPS on a Raspberry Pi 4B system. Also, the device was equipped with voice feedback. In a similar vein, Mukhiddinov and Cho [14] applied deep learning embedded in smart glasses with Detection Transformer (DETR) for object and text recognition functions trained on the Ms COCO, ExDark, and LOL datasets. As a result, the mAP_{50} value was 63.5% for object detection and 92.8% for text detection. The model also handled light intensity and voice feedback. Leong and Ramasamy [15] created smart glasses with object and distance detection based on SSD MobileNet v1 and EfficientDet which were quite light and performed extremely well on the MS COCO and PASCAL VOC datasets. The device comprised audio and vibration feedback controlled via voice commands.

Lee and Cho's scrutiny [16] integrated object recognition, object extraction, outlining, and braille conversion in smart glasses. YOLOv3 [17] was employed to handle multi-class problems at that time [18]. The model was highly efficient with an average braille conversion accuracy of 85% and detection accuracy of 90%. The use of YOLOv3 has proven to be considerably good when applied to smart glasses. As evidence, YOLOv3 was embedded in smart glasses [9, 19, 20], these three studies combined YOLOv3 with voice feedback generated by a text-to-speech (TTS) model. Xia et al. [21] adapted YOLOv3-Tiny for the detection model. The model applied speech commands through the ConvT model combined with a Transducer and Weak-Attention Suppression (WAS). Hence, it was able to provide a good device and user interaction experience. Other results covered obstacle detection, navigation, traffic light detection, NFC payment, emergency calls, and guardian monitoring.

Different from smart glasses in general, Chang et al. [22] proposed IoT-based smart glasses to recognize medicines in pill form. This device was remote via smartphone and integrated with the cloud. The detection model was built by a combination of SSD, ResNet-50, and FPN and voice feedback. As a result, the mAP50 value was 35% with an inference speed of 76 ms/image when trained on the MS COCO dataset. Similar to Chang et al. [22], Li et al. [23] proposed artificial IoT-based and multi-functional smart glasses. This device was equipped with object detection, object distance measurement, and text recognition. YOLOv5 and Optical Character Recognition (OCR) were adopted through the Convolutional Recurrent Neural Networks (CRNN) [24] architecture to reach extremely high accuracy when detecting characters [25]. The results reported that object and text detection accuracy reached 92.16% and 99.91%.

Zhu et al. [26] utilized deep learning and acoustic touch as a substitute for voice feedback in smart glasses. This combination created convenience for users without significantly increasing cognitive load. The acoustic touch technique changed objects entering the field device of view into diverse sound icons [26]. Converting objects into sound icons had been conducted previously through YOLOv5. This model produces better smart glasses than traditional smart glasses.

2.2 Based on sensor

The use of sensors is another alternative to smart glasses. Faster reception of information becomes the primary reason [8]. Busaeed et al. [27] created the LidSonic system which is a machine learning-based device with LiDAR and ultrasonics for visually-impaired people. This system adopted an HC-SR04 ultrasonic sensor, TFmini-s LiDAR, laser, servo, and Bluetooth module connected to an Arduino Uno microcontroller and smartphone. This method offered a cost-effective, easy-to-use solution to enhance mobility and independence for visually impaired people. Bouteraa [28] proposed smart navigation for visually impaired people by adopting a Robot Operating System (ROS), ultrasonic sensors, and LIDAR to detect obstacles. The data were processed by the Raspberry Pi 4 and classified by a fuzzy classifier. The results were conveyed via a haptic and voice interface to the users. The results revealed that this system was effective in navigating indoor and outdoor environments with high accuracy, responsive feedback, and increasing movement independence for visually impaired people.

2.3 Comparison of related research

The development of smart glasses in this paper is based on related research as presented in Table 1. Our research adapted YOLOv10 with additional modifications in the form of a combination of Partial Self-Attention (PSA) and Relative Positional Encoding (RPE) to be applied as an object detection model on our developing device. As a result, it indicates more accurate object detection with support for adaptability. Our model is capable of operating in indoor-outdoor environments with low light intensity. Apart from that, there are navigating and object search modes to be adjusted via voice commands.

Table 1. Comparison of smart glasses models

Author	Model/ Sensor	Object Recognition	Voice Feedback	Speech Command	Object Distance Estimation	Determining Object Location	Low Light Intensity	Indoor	Outdoor	Navigation Mode	Object Search Mode	Ability to Address Contextual Knowledge
[13]	SSDLite MobileNet v2	√	√	√	-	-	-	√	√	√	-	-
[14]	DETR-DC5-R101	√	√	-	-	-	√	√	√	√	-	-
[15]	SSD MobileNet v1 and EfficientDet	√	√	√	√	-	-	√	√	√	-	-
[16]	YOLOv3	√	√	-	-	-	-	√	√	√	-	-
[19]	YOLOv3	√	√	-	-	-	-	√	√	√	-	-
[9]	YOLOv3	√	√	-	√	-	-	√	√	√	-	-
[20]	YOLOv3	√	√	-	√	-	-	√	√	√	-	-
[21]	YOLOv3-tiny	√	√	√	-	-	√	-	√	√	-	-
[22]	SSD-ResNet50-FPN	√	√	-	-	-	-	√	-	-	-	-
[23]	YOLOv5	√	√	-	√	-	-	√	√	√	-	-
[26]	YOLOv5m	√	√	-	√	√	-	√	-	-	√	-
[27]	LiDAR	-	√	√	√	-	√	√	√	√	-	-
[28]	LiDAR	-	√	-	√	√	√	√	√	√	-	-
Our Model	RPSA-YOLOv10	√	√	√	√	√	√	√	√	√	√	√

Received: November 2, 2024. Revised: December 16, 2024.

Implementing the voice feedback feature as an output can enhance the functionality of our smart glasses. Eventually, we have added contextual knowledge handling (e.g. the form of variability in light intensity and object dimensions). This function can increase the adaptability of smart glasses to their environmental context.

3. Proposed method

The smart glasses architecture proposed by us (Fig.1) consisted of two components, namely physical and cyber systems based on processes undertaken at run-time [12, 29]. In the physical system section, interaction occurred between the users and the smart glasses hardware via voice commands, and the results were returned to the users in the form of sounds. Processing commands from users was performed in the cyber system section consisting of three models, namely the detection and distance estimation model, speech recognition model, and text-to-speech (TTS) model. In the speech recognition model, we employed the Speech Recognition module. For the TTS model, we utilized the gTTS module in Python. We designed the detection and distance estimation model with adaptability to handle context uncertainty [30]. In addition, we modified YOLOv10 to boost the performance of the object detection model. The created modifications took place in the attention mechanism section, namely by combining convolution, attention mechanism [31], and feed-forward network (FFN).

The partial Self-Attention module (PSA) in Fig. 2 was designed to overcome computational complexity and high memory usage [11]. After 1x1 convolution, the feature was partitioned into two. Only one part was processed through the NPSA block, namely Multi-Head Self-Attention (MHSA) and FFN. Both were combined and fused with 1x1 convolution. PSA applied BatchNorm which was faster than LayerNorm. It was situated after Stage 4 with the lowest resolution to avoid computational overhead. In this way, PSA enhanced model learning for global representation with low computational cost. Hence, it was able to boost the performance of YOLOv10 [11]. Unfortunately, traditional self-attention in PSA often faced challenges in capturing relative position information between elements in pivotal data for visual tasks. For this reason, we proposed Relative Partial Self-Attention (RPSA) embedded in YOLOv10. In particular, we modified PSA with relative positional encoding (RPE) [32] allowing the model to understand spatial context better with high accuracy. We also added new layers in the form of Dropout and BatchNorm2D for training regularization and stabilization to improve model generalization and faster convergence. Fig. 3 signifies the modified PSA module. First, the data went through 1x1 convolution to change the feature dimensions. Then, it was divided into two equal parts.

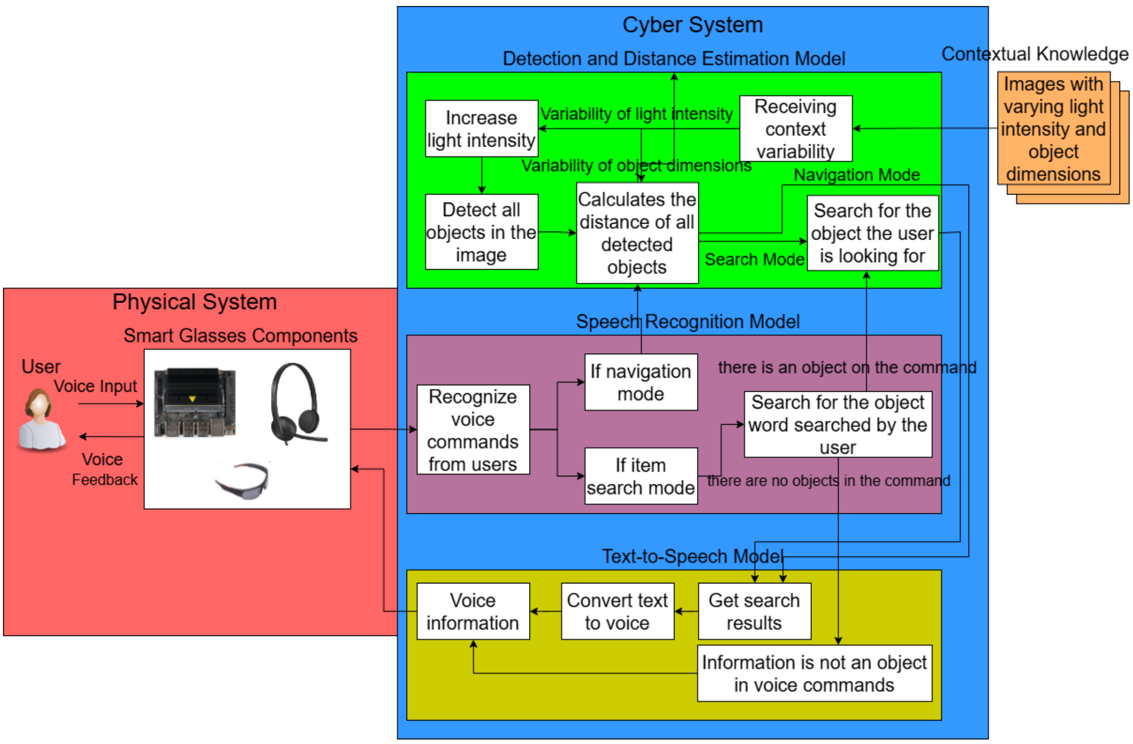

Figure. 1 Smart glasses architecture

Received: November 2, 2024. Revised: December 16, 2024.

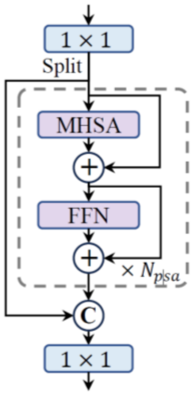

Figure. 2 Partial self-attention module (PSA) [11]

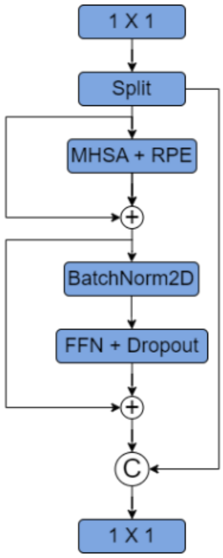

Figure. 3 PSA Improvements on YOLOv10

The first part was processed through MHSA [33] which is equipped with relative positional encoding to understand the spatial context better. The results went to the normalization layer to stabilize the activation distribution. This part was passed to FFN possessing dropout to reduce overfitting. The two parts were recombined before going through a final 1x1 convolution for output. With this structure, PSA had a more efficient and stable global representation. Also, PSA improved model performance. We formulated handling uncertainty by contextual knowledge supported with contextual acceptance before variability enters the machine. As a result, the model could adapt to changing light intensity and calculate approximate object distances under conditions of diverse object dimensions, as represented in Table 2.

The adaptative pattern illustrated in Table 2 is an algorithm aimed at performing context-based adaptation by taking into account the variability

Table 2. Algorithm for receiving contextual knowledge

Smart glasses adaptive pattern
$I \leftarrow C_1, C_2, C_3$
do
let
$I \leftarrow$ inference model
// Monitoring (M) // receiving contextual knowledge
for I in the run-time artifact, do
if (C_2) or (C_1 and C_2) in run-time artifact, then
send information C_2 to analyzer_manager
else if (C_1 and C_3) or (C_1 and C_2 and C_3) in run-time artifact, then
send information C_3 to analyzer_manager
endif
endfor
// Cognition (CG)
for each C_i in analyzer_manager, do
if C_2 is True, then
increase brightness for I
if C_3 is True, then
$O \leftarrow$ predict_object(C_1)
if O is in C_1, then
calculate w and h in O
if w is True, then
calculate D from the camera to O
endif
else
$O \leftarrow$ empty_set
endif
endif
endfor
// Configuration (CF)
if O is empty_set and C_2 is True, then
system $\leftarrow$ increase brightness for I
else if O is not empty_set and LI is True, then
system $\leftarrow$ increase brightness for I
system $\leftarrow$ calculate D from camera to O
system $\leftarrow$ release voice feedback
endif
for each system in run-time artifact, do
send information to M
endfor

contained in the image (I) represented by parameters C_1, C_2, and C_3. The algorithm was adapted from our previous research by focusing on three main components [12]. To illustrate, the first was the Monitoring (M) section where the system detected run-time artifacts (I) to determine whether certain contextual parameters (C_1, C_2, C_3) were detected by setting the rule scheme subsequently:

Rule-1: $if\,(C2 \vee (C1 \wedge C2))$
$then\ send\ C2\ to\ analyzer\_manager$

Rule-2: $if\,\big((C1 \wedge C3) \vee (C1 \wedge C2 \wedge C3)\big)$
$then\ send\ C3\ to\ analyzer\_manager$

Where C1 was the object detected in the image as a result of inference with $C1 = f(I)$ and where

$f(I) = \{O_1, O_2, \dots, O_n\}$, $O_n \in$ detected objects, $C2$ was the low light intensity calculated based on the average pixel intensity using Eq. (1).

$$C2 = \frac{1}{|Int|} \sum_{i=1}^{|Int|} Int_i \tag{1}$$

Where Int was the light intensity value in pixels based on the RGB color channel value, $C3$ was the object dimension in pixels (w, h) based on the size of the bounding box surrounding the detected object. The second stage in this adaptative pattern was the Cognition (CG) part aimed to analyze information from the monitoring stage and take action based on parameter values , $C2$ and, $C3$. If the condition where, $C2$ was sent to the analyzer manager and the, $C2$ value was smaller than the threshold, then the system increased the brightness via Eq. (2).

$$B = C2 \cdot F \tag{2}$$

Where B was the new light intensity value resulting from the increase and F was a multiplier factor which must be $F > 1$. Next, if an object (O) was detected then the system calculated (w, h) dimensions and distance based on the object distance calculation method in Table 3. However, if no object was found then $O = \emptyset$ .. Finally, the adaptative pattern would go through the Configuration (CF) section aimed to execute actions based on the analysis results. Similar to part M, (CF) applied a rules scheme in carrying out its execution with the following rules:

Rule-3: $if\,(O = \emptyset \text{ and } C2 \text{ is } True)$
$then\ system\ increases\ brightness: B \rightarrow I$

Rule-4: $if\,(O \neq \emptyset \text{ and } C2 \text{ is } True)$
$then\ System\ increases\ brightness,$
$calculate\ D\ and\ release\ voice\ feedback$

All information from this CF was sent back to the monitoring stage to ensure real-time system adaptation. In order to clarify our adopted distance calculation method, we display the method in Table 3. Table 3 demonstrates the object distance calculation method extended from [34, 35]. Calculation of object distance required initialization in the form of O. The result of object detection by the system and focal length (f_x, f_y) from the calibration of the camera was adopted. The process began by calculating each dimension of O_i in $C1$ detected by the system. The object dimension value was obtained from the bounding box created for each O_i with corner points marked by tl (top-left), tr (top-right), bl (bottom-left), and br (bottom-left). The

Table 3. Method for calculating object distance

Distance Calculation Pattern

O ← predict_object(C_1)
get the focal length (f_x, f_y) from camera calibration
// Pixel to cm conversion factor

$$f = \frac{pixel\ size\ of\ reference\ object\ the}{original\ size\ of\ reference\ object}$$

// Calculate object dimensions
for O in C1, do
create bounding_box for O_i with point (tl, tr, bl, br)
if O_i has bounding_box, then
determine midpoint:
top:

$$(tpX, tpY) = \left(\frac{tl_0 + tr_0}{2}, \frac{tl_1 + tr_1}{2}\right) \tag{3}$$

bottom:

$$(bmX, bmY) = \left(\frac{bl_0 + br_0}{2}, \frac{bl_1 + br_1}{2}\right) \tag{4}$$

Left:

$$(ltX, ltY) = \left(\frac{tl_0 + bl_0}{2}, \frac{tl_1 + bl_1}{2}\right) \tag{5}$$

right:

$$(rtX, rtY) = \left(\frac{tr_0 + br_0}{2}, \frac{tr_1 + br_1}{2}\right) \tag{6}$$

if midpoint is not empty, then
calculate the height and width of OC_i in pixels:

$$h_{pixel} = \sqrt{(tpX - bmX)^2 + (tpY - bmY)^2} \tag{7}$$

$$w_{pixel} = \sqrt{(ltX - rtX)^2 + (ltY - bmY)^2} \tag{8}$$

calculate the height and width of OC_i in cm:

$$h_{cm} = \frac{h_{pixel}}{f} \tag{9}$$

$$w_{cm} = \frac{w_{pixel}}{f} \tag{10}$$

// Calculate the object distance to camera
if O_i has dimensions, then
if use height for reference, then

$$D = \frac{f_y \times h_{cm}}{h_{pixel}} \tag{11}$$

else if use width for reference, then

$$D = \frac{f_x \times w_{cm}}{w_{pixel}} \tag{12}$$

dimensions of the obtained object were dimensions in 2D space. As a result, there were two values to be searched for, namely height (h) and width (w). These values were gained from each side of the bounding box and the middle value should be identified first via Eq. (2) – Eq. (5). Given this fact, the coordinates of the points on each side of the bounding box were obtained, namely (tpX, tpY) for the top side, (bmX, bmY) for the bottom side, $(ltX, ltY)$for the left side, and (rtX, rtY) for the right side. Next, calculating the height and width of the O_i in pixels (h_{pixel} dan w_{pixel}) with the Euclidean distance equation in Eq. (6) and Eq. (7). (h_{pixel} dan w_{pixel}) should be converted first to cm units to get dimensional values in a real-world representation. Where h_{cm} is the estimated height of the O_i in cm

Table 4. Notations used in the table 3

Variables	Descriptions
O	Predictive results of objects detected by the system based on parameter C1
f_x, f_y	Camera focal length in pixels on the x and y axes as a result of camera calibration.
f	Pixel to cm conversion factor is calculated based on the reference size of the object.
tl, tr, bl, br	Corner points of the bounding box: top-left, top-right, bottom-left, bottom-right.
tpX, tpY	Coordinate of the center point of the top side of the bounding box.
bmX, bmY	Coordinate of the center point of the bottom side of the bounding box.
ltX, ltY	Coordinate of the center point of the left side of the bounding box.
rtX, rtY	Coordinate of the center point of the right side of the bounding box.
h_{pixel}	Object height in pixels calculated through the Euclidean distance formula between the top and bottom sides.
w_{pixel}	The width of the object in pixels, calculated through the Euclidean distance formula between the left and right sides.
h_{cm}	Object height in cm, obtained from h_{pixel} conversion through conversion factor f.
w_{cm}	Object width in cm, obtained from w_{pixel} conversion through conversion factor f.
D	The estimated distance between object O and the camera is calculated through the height or width as a reference.

and w_{cm} is the estimated height of the O_i in cm. After that, h_{cm} and w_{cm} become references for calculating the estimated distance D between O_i and the camera. To clarify all the equations used in Table 3, every variable utilized has been defined in Table 4.

4. Experiment(s)

The experimental instrument in this study adopted Wohlin et al. [36] and extended the research model of Aradea et al. [12, 37, 38]. Table 5 presents the experimental design aimed at developing solutions to problematic variability in uncertain real-world objects characterized by adaptative strategies

Table 5. Experimental Design

No	Descriptions
1	Purposes: a. Developing smart glasses with self-adaptation capabilities in handling contextual uncertainty b. Evaluating the performance of smart glasses
2	Domain: Smart glasses for the visually-impaired people
3	Evaluative Questions (PE): a. PE_1-How do object recognition models handle contextual variability? b. PE_2-What is the performance measure of each element of the object recognition system artifact?
4	Variables (V): a. Response (V_1-failure; V_2- functional and non-functional systems; V_3-new stimulus) b. Measurement of object recognition system performance (V_4-mAP; V_5-inference speed; V_6-object distance estimation; V_7-light intensity)

to enhance performance. Performance encompasses the ability to recognize objects with the RPSA-YOLOv10 model, namely measuring distance based on the variability of object dimensions, light adjustments, user voice commands, and voice notifications for users. Model evaluation was conducted by evaluating mean average precision (mAP), inference speed, frame rate, measuring the estimated distance of the object to the actual distance, and adaptability of light intensity.

On the object recognition model side, we modified the YOLOv10 architecture, specifically in the PSA section to produce a new model in the form of RPSA-YOLOv10 (Fig. 3). As a form of validation, we tested the performance of our model (RPSA-YOLOv10) by training it on the MS COCO dataset to compare with other models with similar datasets. The MS COCO dataset contained approximately 330,000 image data employed for object detection, segmentation, and captioning tasks [39]. Apart from that, MS COCO had 80 classes creating good data diversity. Our experiments were begun by training an object detection or recognition model. To prove that the model improves, we adapted the same hyperparameters as employed by Wang et al. [11]. This hyperparameter consisted of involving 500 epochs, the SGD optimizer, weight decay of 5 × 10-4, learning rate of 10-2, and momentum of 0.9. The training process utilized a GeForce RTX 4090 24 GB GPU with a training time of around 7 days.

Received: November 2, 2024. Revised: December 16, 2024.

Table 6. Comparison of performance of object recognition models

Model	#param	mAP_{50}	mAP_{75}	mAP_{50-95}
YOLOv6m[40]	34.3 M	66.8%	-	49.5%
YOLOv7 [41]	36.9 M	69.7%	55.5%	51.2%
YOLOv8m [42]	25.9 M	66.7%	54.7%	50.2%
YOLOv9m [43]	20 M	68.1%	56.1%	51.4%
YOLOv10m[11]	15.4 M	68.1%	55.8%	51.1%
Our Model	17.4 M	67.3%	54.5%	50.2%

Table 6 deciphers a comparison of our model (RPSA-YOLOv10) with other versions of the YOLO model. In this case, our model underwent degradation in the overall mAP evaluation with mAP50, mAP75, and mAP50-95 values of 67.3%, 54.5%, and 50.2%. This was caused by adaptative performance affecting detection quality. However, by employing a self-attention mechanism designed by combining RPE and PSA, we were able to reduce this impact very well. Furthermore, we compared the performance of our object recognition model with the performance of models applied in various related studies based on the similarity of using datasets, namely, the MS COCO dataset.

The comparative results (Table 7) indicate that our model is very superior to other models although the number of parameters displayed in Fig. 4 remains larger than the SSDLite MobileNet v2 model. In other words, if inference testing is conducted, SSDLite MobilNet v2 [13] will have the highest speed even though it has the worst accuracy. Nevertheless, our model's size is not excessively large, with the number of parameters only slightly exceeding that of the SSDLite MobileNet model. This is demonstrated in the parameter comparison column in Table 7 and the graph in Fig. 4 illustrating

Table 7. Comparison of the performance of object recognition models used in smart glasses in existing research within the related work section, based on the use of the MS COCO dataset.

Author	Model	mAP_{50}	mAP_{50-95}	Number of Classes
[13]	SSDLite MobileNet v2	-	23.4%	80
[14]	DETR-DC5-R101	64.7%	44.9%	80
[16]	YOLOv3	57.9%	33%	80
[22]	Faster-RCNN-ResNet50	-	30%	80
Our Model	RPSA-YOLOv10	67.3%	50.2%	80

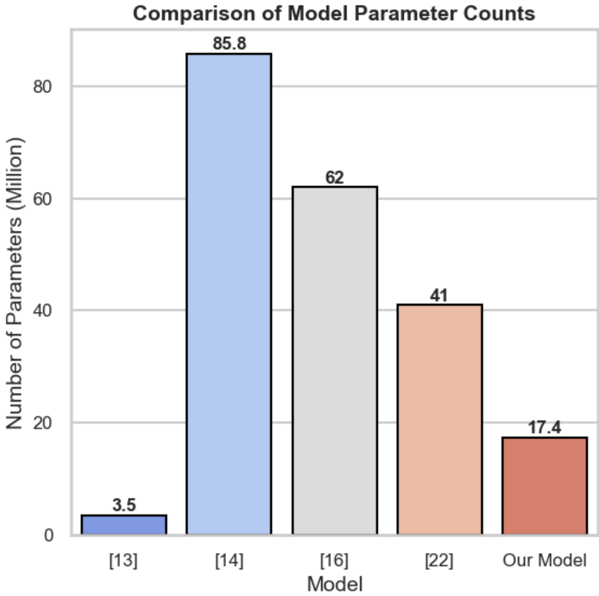

Figure. 4 Comparison of the number of parameters for each model

the parameter comparison. As a result, our model achieves reasonably good inference speed when performing object detection on images. Apart from utilizing benchmark datasets, we also applied custom datasets to enable the system to better recognize the environment where the device was employed. This custom dataset consisted of 1000 images containing 15 object classes.

Table 8 delineates the performance of our model (RPSA-YOLOv10) compared to other versions of YOLO models that can be trained using custom datasets. Testing was performed through the same device, namely the device applied for the RPSA-YOLOv10 training process on the MS COCO dataset. In particular, for testing the mAP metric, our model could produce more value than other YOLO versions with mAP50 at 73.1% and mAP50-95 at 48.3%. Conversely, for inference speed, the performance was lower than YOLOv10. This took place because the size of our model model was slightly larger than the YOLOv10 model [11]. Therefore, it caused the speed of object detection and recognition to operate

Table 8. Performance comparison of various YOLO models trained on custom datasets

Model	mAP_{50}	mAP_{50-95}	Inference Speed
YOLOv3m	61.7%	37.3%	8.3 ms/image
YOLOv5m	61.5%	37.4%	7 ms/image
YOLOv6m	53.7%	32.1%	8.2 ms/image
YOLOv8m	64.3%	40.1%	5.5 ms/image
YOLOv9m	65.4%	40.6%	8.5 ms/image
YOLOv10m	57.9%	35.8%	7 ms/image
Our model	73.1%	48.3%	8.4 ms/image

more slowly. After successfully developing the object recognition model, we subsequently developed adaptative artifacts based on the adaptative patterns in Tables 2 and 3.

The first adaptative feature is calculating the distance of the detected object based on assorted object dimensions. This feature is an extension of our previous papers [37, 38]. The estimated distance calculation undertaken through the adaptative pattern in Table 2 will occur if the conditions $(I = C_1 \wedge C_3) \vee (I = C_1 \wedge C_2 \wedge C_3)$ are met. To validate the adaptability of the designed patterns, we utilized Eq. (12) – (14) to test how accurate the object distance estimation results were. Where dif_i was the value of the difference between the actual distance and the estimated distance in percent, d was the actual distance, d' was the estimated distance, $\bar{\varepsilon_i}$ was the average value of error for one class in one experimental period, n was the number of experiments, ε was the average value of the overall class error in percent and c is the number of classes in the dataset.

$$dif_i = \frac{|d-d'|}{d} \times 100\% \tag{13}$$

$$\bar{\varepsilon_i} = \frac{\sum_{i=0}^{n} dif_i}{n} \tag{14}$$

$$\varepsilon = \frac{\sum_{i=0}^{n} \bar{\varepsilon}_i}{c} \tag{15}$$

Table 9 illustrates the results of testing object distance estimates. The results of estimating object distances at original distances of 0.5 meters, 1 meter, and 2 meters demonstrated that the average error rate for the entire class was 15.95%. On the one hand, the smallest average error was located in the book class with an error of 8%. On the other hand, the largest average error was located in the ladder class with an error of 42.3%. This contributed to a distance estimation accuracy of 84.05%. This value described a fairly high success rate for adaptation. Further, this distance estimation only relied on a camera with only one viewing angle (monocular). Consequently, this test can validate the ability of the system to estimate object distances.

The second adaptative feature is handling low light intensity. The adaptative process operated when the light intensity was below 50 (C2 < 50). This number came from the average value of the RGB color combination in the frame captured by the camera. The increase in new brightness (B_{new}) came from the base value of the old brightness (B_{old}) multiplied by the light intensity multiplier factor (F).

Table 9. Adaptative results for object distance estimates

Object	Original Distance	Estimation	Difference	Average Error
People	0.5 m	0.79 m	58%	29%
	1 m	1.12 m	12%	
	2 m	2.34 m	17%	
Table	0.5 m	0.77 m	44%	16.8%
	1 m	1.04 m	4%	
	2 m	2.05 m	2.5%	
Chair	0.5 m	0.58 m	16%	14.5%
	1 m	0.76 m	24%	
	2 m	1.93 m	3.5%	
Door	0.5 m	0.61 m	22%	16%
	1 m	1.24 m	24%	
	2 m	2.06 m	3%	
Seat	0.5 m	0.63 m	26%	12.8%
	1 m	0.89 m	11%	
	2 m	2.03 m	1.5%	
Plate	0.5 m	0.46 m	8%	6.5%
	1 m	1.03 m	3%	
	2 m	2.17 m	8.5%	
Glass	0.5 m	0.51 m	2%	8.8%
	1 m	1.09 m	9%	
	2 m	1.69 m	15.5%	
Bottle	0.5 m	0.54 m	8%	7.2%
	1 m	1.11 m	11%	
	2 m	2.05 m	2.5%	
Bag	0.5 m	0.65 m	30%	14.3%
	1 m	0.88 m	12%	
	2 m	1.98 m	1%	
Laptop	0.5 m	0.58 m	16%	8.8%
	1 m	1.01 m	1%	
	2 m	1.81 m	9.5%	
Telephone	0.5 m	0.34 m	32%	21%
	1 m	1.11 m	11%	
	2 m	2.4 m	20%	
Television	0.5 m	0.68 m	36%	19.5%
	1 m	1.09 m	9%	
	2 m	2.27 m	13.5%	
Projector	0.5 m	0.37 m	26%	13.8%
	1 m	0.95 m	5%	
	2 m	1.79 m	10.5%	
Book	0.5 m	0.53 m	6%	8%
	1 m	1.06 m	6%	
	2 m	2.24 m	12%	
Stairs	0.5 m	0.85 m	70%	42.3%
	1 m	1.31 m	31%	
	2 m	2.54 m	26%	
Total of Average Error				15.95%

Changes in image brightness could be undertaken with Eq. (1). Table 10 outlines the results of light intensity adaptation based on C2. In particular, the increase in light intensity was always based on the B_{old} value. Thus, it appears that the B_{new} values did

Received: November 2, 2024. Revised: December 16, 2024.

Table 10. Results of adaptation to increasing light intensity

No. Sample	Light Intensity Before (B_{old})	Light Intensity After (B_{new})
1	49.37781944444	118.55763136574
2	46.74598292824	112.22711516204
3	44.08877285879	105.84925231481
4	41.38748003472	99.35508709491
5	40.057916666666	96.16702054398
6	38.721634548611	116.29317650463
7	36.050922743056	108.24479745370
8	33.384502025463	100.20943489583
9	30.723147280093	92.17820486111
10	28.06607378472	112.40079571759
11	26.72938252314	107.06316203703
12	25.37549074074	101.65819849537
13	24.05037442129	96.35608304398
14	22.72607291666	91.06828269675
15	21.41800491898	85.84278067129
16	20.1047265625	80.59685098379
17	18.77662789351	112.75285098379
18	17.44593547453	104.75996788194
19	16.12916059027	96.84081626157
20	14.82116348379	88.97579861111
21	13.50285127314	81.05487760416
22	12.17291608796	73.06952430555
23	10.81183738425	64.90382667824
24	9.46547106481	94.53855815972
25	8.08814525462	80.77312586805
26	6.72029571759	100.82346614583
27	5.35742129629	80.32760271990
28	4.02175636574	80.60934866898
29	2.66026909722	106.51936458333
30	1.32229166666	79.35144560185

not have the same value. This means that when the light intensity increases, the process does not damage the feature information contained in the image. From the test scenario, it was discovered that the average increase in light intensity by the system adaptive process was 328.16% with the final result being an average B_{new} of 95.65.

Fig. 5 showcases the comparative results of B_{old} and B_{new} light enhancement adaptation. To illustrate, the produced a light increase always adjusted to B_{old}. This occurred because of the ability of the system to adapt to the intensity of light received. Hence, it indirectly provided an increase in light without excessively damaging image features. However, this does not guarantee complete protection for images with very low light intensity. Occasionally, this increase in light intensity causes damage to images.

Figs. 6 and 7 are representative samples tested in Table 10. These images indicate examples of images before and after changes in light intensity as a result

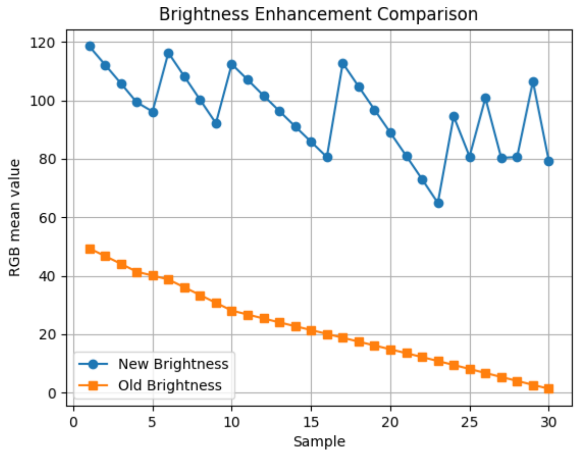

Figure. 5 A Comparative Chart for B_{old} and B_{new}

of system adaptation. As an example, in the 24th sample with Bold < 10 even though it had increased, the results of this increase were not as large as in the 3rd sample. This occurred due to the adjustment of the increase in light intensity based on the Bold value. In other words, the smaller the Bold value, the darker the light intensity would tend to be. This took place to maintain the quality of the images captured by the camera to be readable for the object recognition model.

The final stage is to integrate the previously created model with the speech recognition and text-to-speech (TTS) models. In this case, we adapted Python modules, namely the SpeechRecognition module for speech recognition and the gTTS and

(a)

(b)

Figure. 6: (a) Image of the 3rd sample with a light intensity value of 44,089 and (b) Image of the 24th sample with a light intensity value of 9,465

 DOI: 10.22266/ijies2025.0229.80

Received: November 2, 2024. Revised: December 16, 2024.

(a)

(b)

Figure. 7 System adaptation results: (a) Increase in light intensity in the 3rd sample and (b) Increase in light intensity in the 24th sample

Figure. 8 System detection results

Figure. 9 Utilization of smart glasses

pygame modules for TTS. The integration of this module enabled our smart glasses to have two modes, namely navigation and object search modes. In navigation mode, smart glasses automatically provided a sound notification to users if there was an object 5 meters in front of the camera. In object search mode, smart glasses waited for the instructional voices of users to search for objects. An example of implementing this mode can be viewed in Fig. 8 and Table 11. The notification results contained information on the names of the objects, the directions of the objects based on clockwise directions, and the distances of the objects to the camera. The two modes (navigation and object search) could be switched from each other through voice commands. Further, the system embedded in our smart glasses supports two languages, namely Indonesian and English.

Table 11. Voice commands and notifications

Commands	Voice Notifications
Please find a cup!	The cup is at 7 o'clock. At a distance of 0.42 meters from you.
Find me a book!	The book is at 6 o'clock, at a distance of 0.61 meters from you.
Get me a laptop!	The laptop is at 10 o'clock, at a distance of 0.97 meters from you.
Find me a bottle!	- The bottle is at 8 o'clock, 0.61 meters away from you. - The bottle is at 4 o'clock, 0.71 meters away from you.
Find me a chair!	Please give me another order!

Fig. 9 designates the results of the physical development of smart glasses. The applied main components consist of a microcontroller, glasses with a camera, and headphones. We utilized Jetson Nano with 128-core NVIDIA Maxwell™ architecture GPU specifications, Quad-core ARM® Cortex®-A57 MPCore processor, and AI performance of 472 GFLOPS. As a result, it is appropriate with every process executed by the device. Apart from that, we employed a camera embedded in the glasses with 8 MP specifications to stream video as input to the system. Each of these components supported the performance of the SACPS adaptative system designed in Fig. 1. With this in mind, it can be an alternative solution for visually impaired people to undergo their daily activities more comfortably.

5. Conclusion

This study has successfully developed a model for smart glasses based on the Self-Adaptive Cyber Physical System (SACPS) artifact [12]. The model is formulated with object recognition capabilities and adaptability regarding calculating object distance, and light intensity, receiving voice commands, and issuing voice notifications. The object recognition model embedded in the device is the result of a modification of the YOLOv10m model [11] named RPSA-YOLOv10 and an extension of the model [37, 38]. In some test scenarios, our object recognition model performs better than previous models. Likewise, our model has adaptability designed based on contextual knowledge to handle context uncertainty. The speech recognition and text-to-speech modules add convenience and functionality where these smart glasses can be activated via voice commands through two modes, namely navigation and object search modes.

Conversely, behind the attained success, there are still shortcomings. As an example, although the mAP value is better than the previous version of the model, the RPSA-YOLOv10 model signifies an increase in the number of parameters supporting it heavier than other models. Moreover, the limitations of calculating object distances with only a monocular camera make distance estimates less accurate due to limited viewing angles. Future work is expected to be able to overcome this problem by modifying the architecture without leaving side effects on other elements. By doing so, it can support the concept of Green AI. The use of other object distance estimation techniques requires to take into account to boost accuracy in predicting object distances better.

Conflicts of Interest

The authors declare that there is no conflict of interest in this paper.

Author Contributions

Conceptualization, Aradea and Ghatan Fauzi Nugraha; methodology, Aradea; formal analysis, reviewing & supervision, Aradea; Investigation, Aradea; formal analysis, Aradea; data curation, Irfan Darmawan, validation, Irfan Darmawan and Rianto; Resources, Rianto, writing—review and editing, Rianto; Investigation, Rianto; software, Ghatan Fauzi Nugraha; visualization, Ghatan Fauzi Nugraha; writing—original draft preparation, Ghatan Fauzi Nugraha.

Acknowledgments

The present study is supported by the Institute for Research and Community Services of Siliwangi University, and the Research Group of Intelligent Systems and Informatics (ISI) at the University of Siliwangi. Likewise, this study is a manifestation of the strategic program of the Ministry of Research, Technology and Higher Education of the Republic of Indonesia (No. No.289/UN58.06/PM.00.00/2024) related to the research developments in Indonesia.

References

[1] J. Wang, S. Wang, and Y. Zhang, "Artificial intelligence for visually impaired", *Displays*, Vol. 77, No. February, p. 102391, 2023, doi: 10.1016/j.displa.2023.102391.

[2] R. R. A. Bourne *et al.*, "Causes of blindness and vision impairment in 2020 and trends over 30 years, and prevalence of avoidable blindness concerning VISION 2020: The Right to Sight: An analysis for the Global Burden of Disease Study", *Lancet Glob Health*, Vol. 9, No. 2, pp. e144-e160, 2021, doi: 10.1016/S2214-109X(20)30489-7.

[3] K. C. Shahira and A. Lijiya, "Towards Assisting the Visually Impaired: A Review on Techniques for Decoding the Visual Data from Chart Images", *IEEE Access*, Vol. 9, pp. 52926-52943, 2021, doi: 10.1109/ACCESS.2021.3069205.

[4] M. Mashiata *et al.*, "Towards assisting visually impaired individuals: A review on the current status and prospects", *Biosens Bioelectron X*, Vol. 12, No. August, p. 100265, 2022, doi: 10.1016/j.biosx.2022.100265.

[5] H. Walle, C. De Runz, B. Serres, and G. Venturini, "A Survey on Recent Advances in AI and Vision-Based Methods for Helping and Guiding Visually Impaired People", *Applied Sciences (Switzerland)*, Vol. 12, No. 5, 2022, doi: 10.3390/app12052308.

[6] G. F. Nugraha, Aradea, Rianto, and R. Mardiati, "Object Detection for the Visually Impaired: A Systematic Literature Review", In: *Proc. of 2024 10th International Conference on Wireless and Telematics (ICWT)*, pp. 1-16 2024, doi: 10.1109/ICWT62080.2024.10674727.

[7] M. M. Valipoor and A. de Antonio, "Recent trends in computer vision-driven scene understanding for VI/blind users: a systematic mapping", *Univers Access Inf Soc*, Vol. 22, No. 3, pp. 983-1005, 2023, doi: 10.1007/s10209-022-00868-w.

[8] K. Manjari, M. Verma, and G. Singal, "A survey on Assistive Technology for visually impaired", *Internet of Things (Netherlands)*, Vol. 11, 2020, doi: 10.1016/j.iot.2020.100188.

[9] R. C. Joshi, S. Yadav, M. K. Dutta, and C. M. Travieso-Gonzalez, "Efficient multi-object detection and smart navigation using artificial intelligence for visually impaired people",

Entropy, Vol. 22, No. 9, 2020, doi: 10.3390/e22090941.

[10] M. Noman, V. Stankovic, and A. Tawfik, "Portable offline indoor object recognition system for the visually impaired", *Cogent Eng*, Vol. 7, No. 1, 2020, doi: 10.1080/23311916.2020.1823158.

[11] A. Wang *et al.*, "YOLOv10: Real-Time End-to-End Object Detection", *arXiv preprint arXiv:2405.14458*, pp. 1-18, 2024, [Online]. Available: http://arxiv.org/abs/2405.14458

[12] Aradea, I. Darmawan, Rianto, H. Mubarok, and G. F. Nugraha, "Object Recognition Model for Smart Glasses Based on Self-Adaptive Cyber-Physical System", *Technical Report, Research Group of the Intelligent Systems and Informatics (ISI), Department of Informatics, Siliwangi University*, 2023.

[13] R. Bin Islam, S. Akhter, F. Iqbal, M. Saif Ur Rahman, and R. Khan, "Deep learning based object detection and surrounding environment description for visually impaired people", *Heliyon*, Vol. 9, No. 6, p. e16924, 2023, doi: 10.1016/j.heliyon.2023.e16924.

[14] M. Mukhiddinov and J. Cho, "Smart glass system using deep learning for the blind and visually impaired", *Electronics (Switzerland)*, Vol. 10, No. 22, 2021, doi: 10.3390/electronics10222756.

[15] X. Leong and R. Kanesaraj Ramasamy, "Obstacle Detection and Distance Estimation for Visually Impaired People", *IEEE Access*, Vol. 11, pp. 136609-136629, 2023, doi: 10.1109/ACCESS.2023.0322000.

[16] D. Lee and J. Cho, "Automatic Object Detection Algorithm-Based Braille Image Generation System for the Recognition of Real-Life Obstacles for Visually Impaired People", *Sensors*, Vol. 22, No. 4, 2022, doi: 10.3390/s22041601.

[17] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement", 2018, [Online]. Available: http://arxiv.org/abs/1804.02767

[18] J. Terven, D. M. Córdova-Esparza, and J. A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS", *Mach Learn Knowl Extr*, Vol. 5, No. 4, pp. 1680-1716, 2023, doi: 10.3390/make5040083.

[19] J. Y. Lin, C. L. Chiang, M. J. Wu, C. C. Yao, and M. C. Chen, "Smart Glasses Application System for Visually Impaired People Based on Deep Learning", In: *Proc. of Indo-Taiwan 2nd International Conference on Computing, Analytics and Networks, Indo-Taiwan ICAN 2020 - Proceedings*, pp. 202-206, 2020, doi: 10.1109/Indo-TaiwanICAN48429.2020.9181366.

[20] G. D. V Cortez, J. C. D. Valenton, and J. B. G. Ibarra, "Low-Cost Smart Glasses for Blind Individuals using Raspberry Pi 2", *Journal of Positive School Psychology*, Vol. 6, No. 3, pp. 6081-6088, 2022.

[21] K. Xia, X. Li, H. Liu, M. Zhou, and K. Zhu, "IBGS: A Wearable Smart System to Assist Visually Challenged", *IEEE Access*, Vol. 10, pp. 77810-77825, 2022, doi: 10.1109/ACCESS.2022.3193097.

[22] W. J. Chang, L. B. Chen, C. H. Hsu, J. H. Chen, T. C. Yang, and C. P. Lin, "MedGlasses: A wearable smart-glasses-based drug pill recognition system using deep learning for visually impaired chronic patients", *IEEE Access*, Vol. 8, pp. 17013-17024, 2020, doi: 10.1109/ACCESS.2020.2967400.

[23] J. Li *et al.*, "An AIoT-Based Assistance System for Visually Impaired People", *Electronics (Switzerland)*, Vol. 12, No. 18, 2023, doi: 10.3390/electronics12183760.

[24] G. Keren and B. Schuller, "Convolutional RNN: an Enhanced Model for Extracting Features from Sequential Data", *ArXiv preprint arXiv: 1602.05875*, 2016. Available: http://arxiv.org/abs/1602.05875

[25] Y. Jiang, H. Dong, and A. El Saddik, "Baidu Meizu Deep Learning Competition: Arithmetic Operation Recognition Using End-to-End Learning OCR Technologies", *IEEE Access*, Vol. 6, pp. 60128-60136, 2018, doi: 10.1109/ACCESS.2018.2876035.

[26] H. Y. Zhu *et al.*, "An investigation into the effectiveness of using acoustic touch to assist people who are blind", *PLoS One*, Vol. 18, No. 10 pp. 1-24, 2023, doi: 10.1371/journal.pone.0290431.

[27] S. Busaeed, R. Mehmood, I. Katib, and J. M. Corchado, "LidSonic for Visually Impaired: Green Machine Learning-Based Assistive Smart Glasses with Smart App and Arduino", *Electronics (Switzerland)*, Vol. 11, No. 7, 2022, doi: 10.3390/electronics11071076.

[28] Y. Bouteraa, "Smart real-time wearable navigation support system for BVIP", *Alexandria Engineering Journal*, Vol. 62, pp. 223-235, 2023, doi: 10.1016/j.aej.2022.06.060.

[29] M. K. Habib and C. Chimsom I, "CPS: Role, Characteristics, Architectures and Future Potentials", *Procedia Comput Sci*, Vol. 200, pp. 1347-1358, 2022, doi: 10.1016/j.procs.2022.01.336.

[30] E. Zavala, X. Franch, J. Marco, A. Knauss, and D. Damian, “SACRE: Supporting contextual requirements’ adaptation in modern self-adaptive systems in the presence of uncertainty at runtime”, *Expert Syst Appl*, Vol. 98, pp. 166-188, 2018, doi: 10.1016/j.eswa.2018.01.009.

[31] D. Bahdanau, K. H. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate”, In: *Proc. of 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp. 1-15, 2015.

[32] P. Shaw, J. Uszkoreit, and A. Vaswani, “Self-attention with relative position representations”, In *Proc. of 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Vol. 2, pp. 464-468, 2018, doi: 10.18653/v1/n18-2074.

[33] A. Vaswani *et al.*, “Attention is all you need”, *Adv Neural Inf Process Syst*, Vol. 2017-Decem, No. Nips, pp. 5999-6009, 2017.

[34] A. Rosebrock, “Measuring the size of objects in an image with OpenCV”, pyimagesearch. Accessed: Jul. 28, 2024. [Online]. Available: https://pyimagesearch.com/2016/03/28/measuring-size-of-objects-in-an-image-with-opencv/

[35] A. Rosebrock, “Find the distance from camera to object/marker using Python and OpenCV”, pyimagesearch. Accessed: Feb. 10, 2024. [Online]. Available: https://pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/

[36] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. doi: 10.1007/978-3-642-29044-2.

[37] Aradea, I. Supriana, and K. Surendro, “ARAS: adaptation requirements for adaptive systems”, *Automated Software Engineering*, Vol. 30, No. 1, p. 2, 2023, doi: 10.1007/s10515-022-00369-3.

[38] A. Aradea, R. Rianto, and H. Mubarok, “Inference Model for Self-Adaptive IoT Service Systems”, *International Journal of Intelligent Engineering and Systems*, Vol. 14, No. 4, pp. 337-349, 2021, doi: 10.22266/ijies2021.0831.30.

[39] T. Y. Lin *et al.*, “Microsoft COCO: Common objects in context”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 8693 LNCS, No. PART 5, pp. 740-755, 2014, doi: 10.1007/978-3-319-10602-1_48.

[40] C. Li *et al.*, “YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications”, 2022, [Online]. Available: http://arxiv.org/abs/2209.02976

[41] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors”, In: *Proc. of 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, pp. 7464-7475, 2023, doi: 10.1109/cvpr52729.2023.00721.

[42] G. Jocher, A. Chaurasia, and J. Qiu, “YOLOv8 by Ultralytics”, *Scientific Research*, 2023. Accessed: Feb. 02, 2024. [Online]. Available: https://github.com/ultralytics/ultralytic

[43] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, “YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information”, *arXiv:2402.13616*, pp. 1-18, 2024, [Online]. Available: https://github.com/WongKinYiu/yolov9.