Paper 05 / 2022 doi: 10.7795/320.202205

JUNGE wissenschaft

JungforscherInnen publizieren
online | *peer reviewed* | original

Verlag:
Physikalisch-
Technische
Bundesanstalt
PTB

Technik

Wearable assistant guiding through traffic

Development and optimization of a portable guidance system for the blind and visually impaired with the application of artificial intelligence

This work describes a new type of portable, self-regulating guidance system, which learns to recognize obstacles with the help of a camera, artificial intelligence, and various sensors and thus warn the wearer through audio signals. It will be described how the individual components are constructed, how they perform in real-life applications, and how those affected commented on the final device.

DER JUNGFORSCHER

Tamas Nemes (2003),
Gymnasium der Regensburger Domspatzen

Eingang der Arbeit:
18.6.2021
Arbeit angenommen:
25.8.2021

Wearable assistant guiding through traffic

Development and optimization of a portable guidance system for the blind and visually impaired with the application of artificial intelligence

1. Introduction

1.1 Summary

The aim of this work is to develop a fully functional and user-friendly prototype for an AI-based guidance system for the blind called GUIDE-Walk, which should support the inclusion of visually impaired people, and to further develop concepts on how to turn it into a product suitable for mass production. The device is based on a previous prototype model built in 2019 [1], which was equipped with basic functions such as distance measuring and real-time inference with a pre-trained object detection model. Although many measures have been taken to support the inclusion of the visually impaired in everyday traffic, those affected are still exposed to unexpected events and unknown obstacles in yet unfamiliar areas. Small digital guidance systems can support an expansion of self-reliance for its users, making them more aware of their surroundings and independent from costly methods like guide dogs. The proposed guidance system uses a camera with an object detection network to identify potential dangers and relevant objects in road traffic and notifies the blind wearer via audio warnings. It further collects and uses additional information from a distance and a motion sensor to optimize performance. By receiving this processed information about objects in the immediate vicinity and in the distance alike as well as about how far they are away, blind users can get a better picture of their surroundings including objects they otherwise would not have noticed. The device is hung around the neck using a loop and can be controlled via tilting gestures. A special focus lies on the following aspects:

- Focusing on a limited number of core functions and preferring quality over quantity
- Keeping the components' complexity at a minimum to ensure low costs in future industrial production
- Implementation of the feedback from test subjects

Code is available at: https://github.com/Totemi1324/GUIDE-Walk-v2.0https://github.com/Totemi1324/GUIDE-Walk-v2.0

1.2 Related Work

With artificial intelligence (AI) being an indispensable part of today's society, many approaches to automated blind guidance systems have sprung up in recent years. In the early 2010s, the first working approaches were created, [2] and [3] just being two examples. These rely on the analysis of camera images on the pixel level and the determination of key points with stereoscopic camera techniques. The algorithms can detect paths as well as moving objects in general. However, due to the lack of computing power and fewer advancements in the AI field at that time, mostly arithmetic methods were utilized instead of AI. Hard-coded calculations like these are case-specific and very sensitive to little changes in the environment such as different types of asphalt or weather conditions. After massive progress in artificial intelligence and its efficiency, papers such as [4] demonstrated approaches with utilizing image classification to recognize relevant scenes. The referenced paper uses a CIFAR-10 classification model on a Raspberry Pi to categorize an image into 10 classes, such as "car" or

doi: 10.7795/320.202205

"building". Although they provide a far wider spectrum of accuracy and reliability, they are of limited use in guidance systems since classification networks are not capable of locating and identifying the specific situations of obstacles, which means blind users have no perception of depth. Further, in an attempt to replace conventional guide dogs, robotic alternatives like [5] were invented, a dog-like robot on a leash that scans the environment using a 2D LiDAR. The work has a great focus on machine-human interaction, an aspect very important in a guidance system one has to trust in. [6] and [7] are other robotic approaches that integrated this technology directly into assistive devices already in use, such as walkers or white canes. Both use pattern-recognition algorithms to detect objects and estimate poses. With Edge AI coming up as a future technology, companies like Microsoft® also designed free smartphone apps like SeeingAI™ [8], a very advanced program for recognizing faces and scenes, reading text and barcodes, and even perceiving color. The inference happens through the smartphone camera, which requires the user to hold the device steadily. Only a small minority of these ideas were developed into marketable devices, examples being OrCam® MyEye™ [9] or InnoMake™ [10], today's most developed assistance systems attachable to glasses or shoes. These devices cost between 2,800 to 5,800 $, thus limiting the number of affected people who can afford them drastically.

2. Concept and Structure

2.1 Basic Concept

The focus of this research was to develop a device that combines suitability for everyday use with high functionality and good performance and can therefore be turned into a product capable of competing with blind guidance devices that are on the market today. The prototype from 2019 [1] was based entirely on Python code, which provided an easy use to start, but in comparison, it is very inefficient and hardly suitable for mass production.

Therefore, this concept should be realized using C++ code only. The main advantages of C++ over Python are a shorter runtime and better control over memory management, which is why these aspects are primarily optimized in the code. Furthermore, C++ provides many opportunities for the usage of external libraries, e.g. OpenCV for image processing and Boost for logging, since most of these widely used applications already utilize C++ in the backend. Finding and implementing suitable libraries for all functionalities and components accounts for a great part of software development.

Fig. 1 illustrates all the components of the device prototype with a corresponding designation.

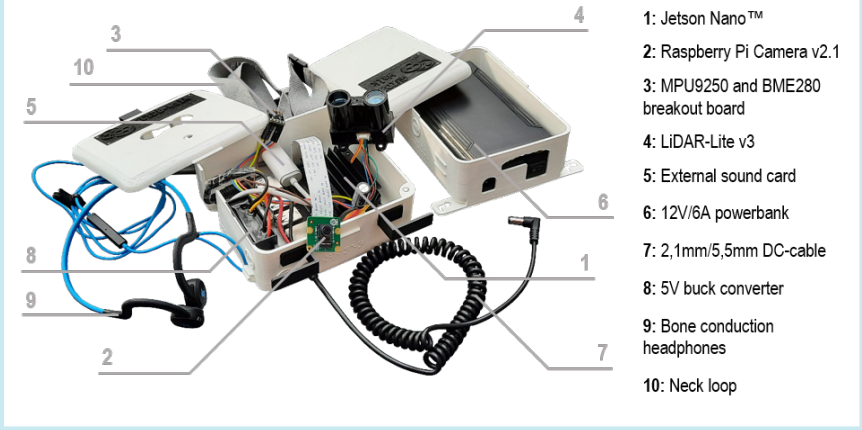

Fig. 1: Components of the prototype, numbered

2.2 Theoretical Foundations for Object Detection Networks

As indicated, AI plays a crucial role in this application, and all other components were designed to be centered around and complement it in a meaningful way. To better understand the functionality of the AI model used in the program, the structure with the key components will be briefly explained below.

An object detection network, as it has already been used in the 2019 prototype [1], suits the application best, as it can detect the position of relevant objects in the image with bounding boxes. The requirements were that the network should be as fast and, at the same time, as accurate as possible. Additionally, it should be controllable via C++. After researching the AI model mentioned before, the SSD architecture by Wei Liu et al. proved to be most suitable, which is explained in detail in [11] and combines the properties referred to above.

SSD stands for "Single Shot MultiBox Detector" and describes a type of Convolutional Neural Network (CNN) that does not use fully-connected layers and, therefore, only needs one pass to make detections. It consists of two main parts: the support network and the SSD add-on. The latter is common to all SSD models, while the former can theoretically be replaced by any network. The structure is shown in Fig. 2.

The support network, which is VGG-16 in Fig. 2, is a slightly adapted CNN and largely determines the speed and accuracy of the entire network. For this application, MobileNetV2, the second iteration of the network originally described in [12], has a good balance between accuracy, model size, and number of operations, which are important for the use on edge devices.

doi: 10.7795/320.202205

These numbers can be observed in Tab. 1, which was adapted from [12] and shows the competitive ImageNet accuracy with few parameters. MobileNetV2 is a relatively new and fast CNN made up of convolution and pooling layers and was specifically created for mobile vision applications. Its task is to generate a feature map from the input, a matrix that summarizes the results of all pattern comparisons of the convolution layers. Then the SSD addition divides the output of the support network into a grid, the resolution of which becoming smaller and smaller as the inferencing proceeds. The innovative aspect of SSD is that a certain number of bounding boxes with pre-determined aspect ratios are created in each grid cell. Layer Conv4_3, for example, divides the output into a grid of size 38 × 38, and makes four predictions in each cell (hence "MultiBox"), thereby creating a fixed number of boxes (N_o = number of output boxes, b_c = boxes per cell, s = grid size):

$$N_o = b_c \cdot s^2 = 4 \cdot 38^2 = 5776 \quad (1)$$

Summing up the predictions made by each layer, the final network output yields a total of 8,732 detections per class as seen in Fig. 2. Therefore, a non-maxima suppression is necessary to filter duplicates or boxes that probably belong to the same object. Finally, the detections are sorted according to confidence, outputting the top 100.

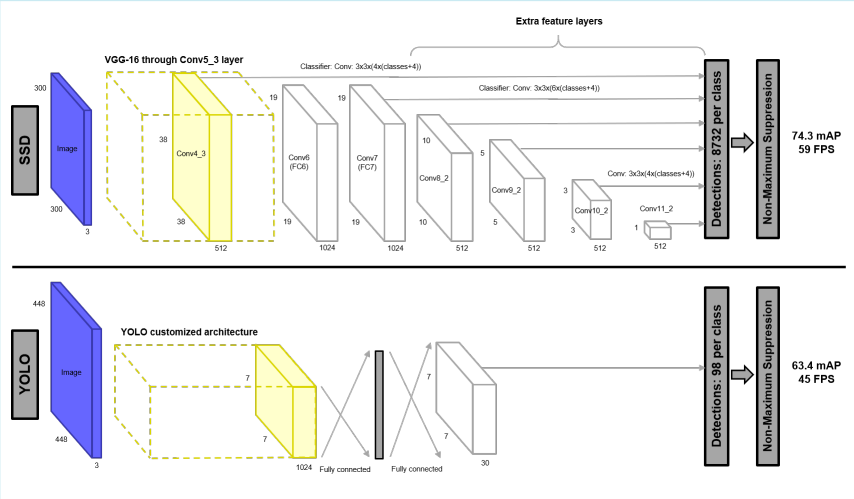

Fig. 2: Schematic of SSD in comparison to the state-of-the-art network YOLO, adapted from [11].

3. Method

3.1 Training and Implementation of the Object Detection Network

One of the main reasons for the decision in favor of MobileNet-SSD v2 was its C++ compatibility. An implementation of the network for the deep learning framework Caffe can be found on GitHub [13], which is also C-based and whose networks can be used for inference with the lightweight OpenCV DNN module (*net.h*).

The network should be able to recognize ten different objects in the images captured by the camera: pedestrians, cars, buses, bicycles, motorcycles, benches, chairs, garbage bins, and red and green pedestrian lights. To ensure the best possible performance, it is necessary to train the network with these classes only. But since there is no ready-to-use dataset just for these classes, one had to be created. Using custom Python scripts, the images of the required classes and the annotations from the MS-COCO [14] and Pascal-VOC [15] datasets were extracted, converting them into the XML format. Also, the Ampelpilot [16] dataset was added, created by teams of the Hochschule Augsburg and the University of Tübingen for recognizing German pedestrian traffic lights (which was relevant because of the different appearance of American models). Finally, own recordings were contributed by sorting and labeling them with the LabelImg tool. In total, 46,512 pictures came together, of which 24,686 are from COCO and VOC and 21,826 are own ones (total size: approx. 8GB). The distribution of classes among the images is shown in Fig. 3. Both the Python code and the dataset, as well as the AI configuration files, can be viewed and downloaded from the GitHub repo.

For pre-training and finetuning, the Caffe framework was used on a high-performance computer (CPU: AMD® Ryzen 9 3950X with 32 threads; GPU:

Tab. 1: MobileNet in comparison to popular models, adapted from [12].

Model	ImageNet accuracy in %	Million Mult-Adds	Million parameters
MobileNet-224	70.6	569	4.2
GoogleNet	69.8	1550	6.8
VGG-16	71.5	15300	138

doi: 10.7795/320.202205

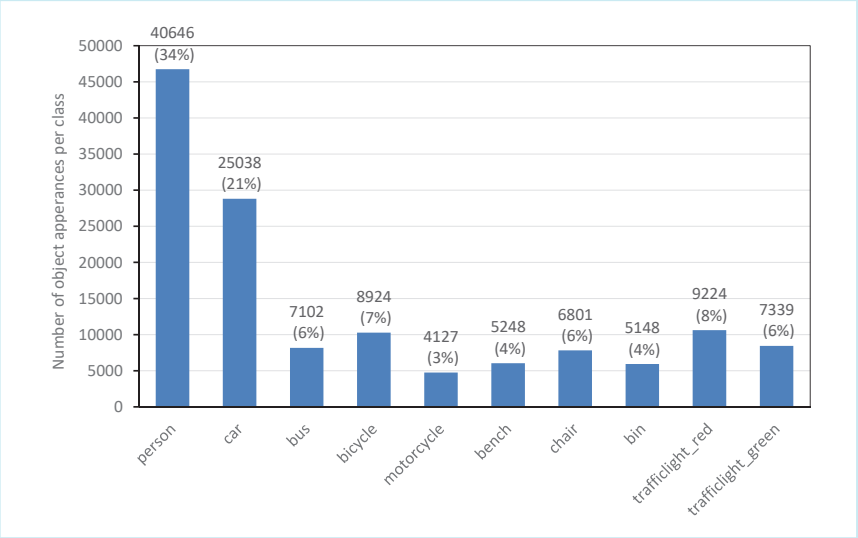

Fig. 3: Total number of object appearances per class in the final dataset

Nvidia® RTX 3070; RAM: 32GB). This computing capacity was able to achieve consistent results. By applying a training split of 0.85 % for the *train-val* and 0.15 % for the *test* dataset, the network was trained with 39,535 and tested with 6,977 images. The separation is done automatically by scripts to ensure that the test set does not contain images from the training set.

3.2 Gesture Recognition with a Motion Sensor

In addition to the AI, which makes up the core of the device, several sensors are used to make the object detection more secure and to extend it with additional functionality. A GPIO extension header with 40 pins is integrated on the Jetson Nano™ board, to which the sensors can be connected. It has two I²C, two SPI, and one UART bus and is almost identical to that of the Raspberry Pi in terms of pin assignment.

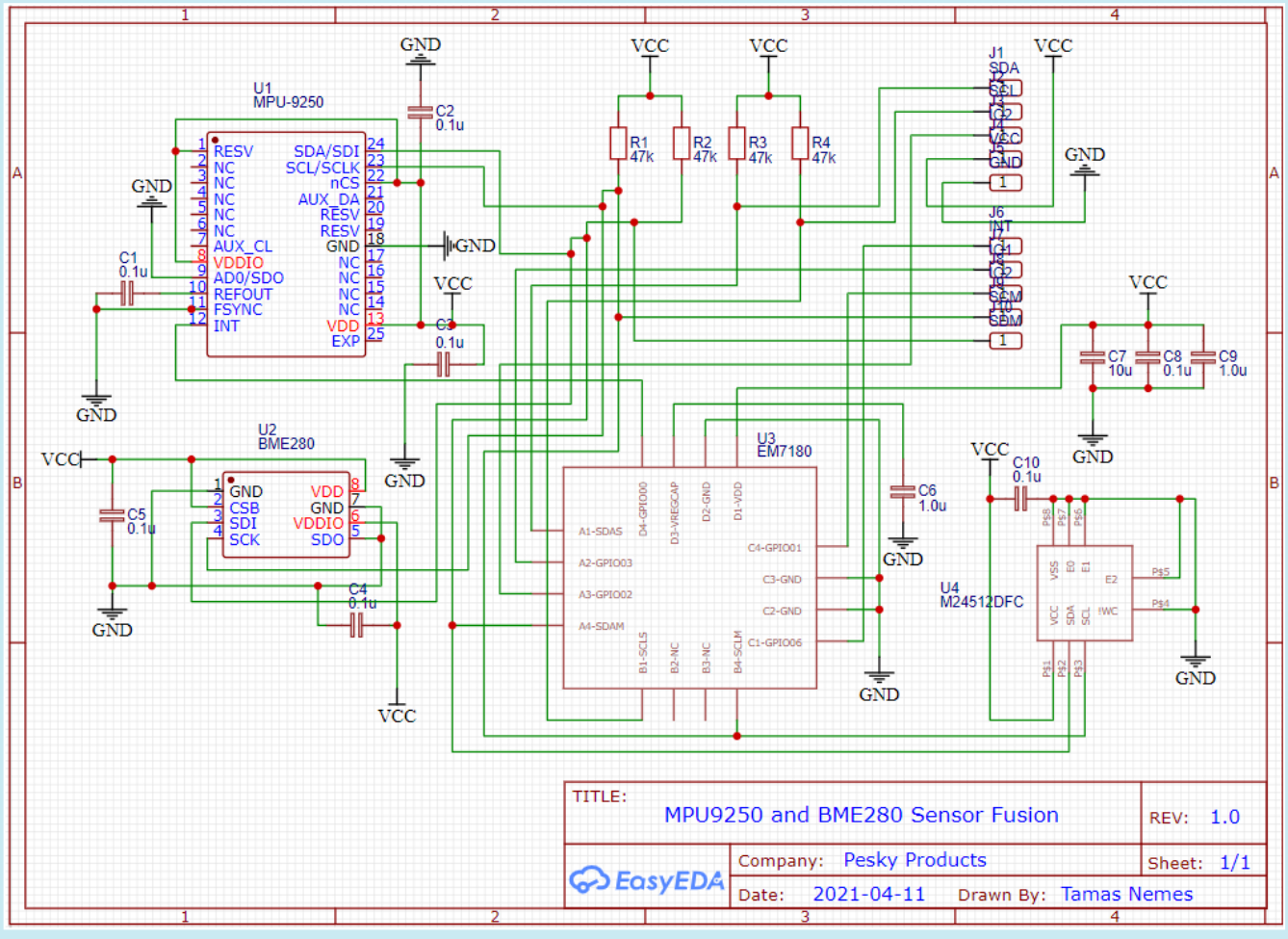

Fig. 4: Schematic sheet of the MPU9250 and BME280 sensor fusion

doi: 10.7795/320.202205

To increase the options for detecting movement on the road, a motion sensor was integrated into the device's housing. The sensor was required to be C++ compatible and be easily accessible using the Jetson Nano™'s integrated libraries for communication via GPIO pins. The 3-axis sensor MPU9250 by InvenSense® with integrated accelerometer, gyroscope, and magnetometer and the BME280 temperature and pressure sensor by Bosch® are to this day the most commonly used and compact-sized integrated circuits (ICs) for this purpose. They are connected with jumper cables over one I²C bus only using a breakout board with an additional EM7180 motion co-processor and an M24512DFC EEPROM from Tindie (*usfs_master.h*). In Fig. 4, the connection of the components can be seen in detail.

With the measurements retrieved from the breakout board, additional data can be calculated. The primary data output from the EM7180 are quaternions (q_w, q_x, q_y, and q_z), which uniquely define device orientation and can be converted to the Euler angles roll (formula 2), pitch (formula 3), and yaw (formula 4) to determine the absolute orientation in space, as proved by [17].

The BME280 sensor also returns data on atmospheric pressure, from which the altitude above sea level can be estimated with the formula (5) derived from the International Standard Atmosphere defined in [18]. p denotes the measured pressure in hPa and p_0 is equal to the pressure at sea level (1013.25 hPa):

$$h = \left[1 - \left(\frac{p}{p_0}\right)^{\frac{1}{5.255}}\right] \cdot 44330 \qquad (5)$$

The sensors are applied for a wide variety of use cases in the device. The idea of programming tap detection already existed in the prototype phase to enable the user to interact intuitively with the device through tapping gestures, for example, to call up information about the surroundings. After gathering sample data and plotting them in graphs, it became evident that the MPU9250 accelerometer is not suitable for a reliable tap detection algorithm since the measured values were impaired by unintentional movements and extreme noise during walking. Therefore, gesture control was solved with the Euler angles instead. The wearer can turn the device in different angles and use it to perform four gestures: tilt to the right for information on time, temperature and altitude, to the left to activate standby mode, and up to switch off the device, which must be executed twice to confirm it. Furthermore, the distance feedback can be deactivated with a downward tilt. A gesture is registered whenever a certain threshold value for the alignments is exceeded, which was chosen as 60 degrees in any direction from the idle state.

One of the most important functions of motion detection is to distinguish between walking and standing. This is used for constraining object recognition: pedestrian traffic lights are only recognized when the wearer stops to minimize false recognitions and thus to ensure more safety. To achieve this, the last 15 measurement values of the accelerometer are recorded in an array from which the standard deviation is calculated. This term describes how much the individual values deviate from their average and therefore indicates walking. Since the values are individual samples,, the formula (6) for the sample standard deviation distribution [19] is used:

$$s = \sqrt{\frac{1}{n-1} \cdot \sum_{i=1}^{n} (x_i - \bar{x})^2} \qquad (6)$$

When walking, the spread of the measured values is relatively large, but if it falls below a threshold value, standing is assumed. The mean value of the height above sea level at two different points in time can be used to determine and warn of a sloping surface. Lastly, the camera's images are cropped using the y-axis of the gyroscope, which measures three-dimensional changes in angle. When the wearer rotates, the parts of the image that it rotates away from are insignificant and can be removed to optimize the performance of the AI. The amount of image that gets cropped depends on the rotation speed. All evaluations proved to be accurate after some minor optimizations.

3.3 Distance Measurement

The distance sensor, namely an ultrasonic sensor, was already an integral part of the original prototype. This served the purpose of compensating for the weaknesses of the AI object detection by recognizing not trained obstacles in the immediate vicinity. Similar to parking aids, the

$$\phi = \arctan 2\left[2(q_w \cdot q_x + q_y \cdot q_z), q_w^{\,2} - q_x^{\,2} - q_y^{\,2} + q_z^{\,2}\right] \cdot \frac{180}{\pi}$$

Formula (2)

$$\theta = -\arcsin\left[2(q_x \cdot q_z - q_w \cdot q_y)\right] \cdot \frac{180}{\pi}$$

Formula (3)

$$\psi = \arctan 2\left[2(q_x \cdot q_y + q_w \cdot q_z), q_w^{\,2} + q_x^{\,2} - q_y^{\,2} - q_z^{\,2}\right] \cdot \frac{180}{\pi} + 13.8$$

Formula (4)

doi: 10.7795/320.202205

program should continuously measure the distance and give the wearer feedback through repeated signal tones as to whether they are approaching an obstacle and its distance from the wearer.

However, the ultrasonic sensor (HC-SR04 by Sparkfun®) used in the prototype is struggling with significant accuracy problems on the Jetson system, which made it impossible to provide reliable distance feedback. While the functionality of the manufacturer's Arduino library can be transformed to run on the Jetson Nano™, its execution requires superuser privileges and is much less precise because of the lags and delays caused by the operating system. After evaluating different sensors, the LiDAR Lite v3 by Garmin® has been chosen, which is a Time-of-Flight sensor with I^2C communication and one of the cheaper LiDAR sensors on the market. It measures distances with a laser by determining the time between the pulse of the transmitter and the impact of the light on the detector. The instructions manual denotes a measurement accuracy of ±2.5 cm (≈0.9 in) and a range of ~40 m (≈131.2 ft) for the breakout module, which has a C++ interface directly from the manufacturer (*lidar.h*). For future versions of the device, inexpensive LiDAR chips can be utilized such as the VL53L5 by ST®, which can also measure in a two-dimensional surface area by emitting an 8 × 8 laser grid.

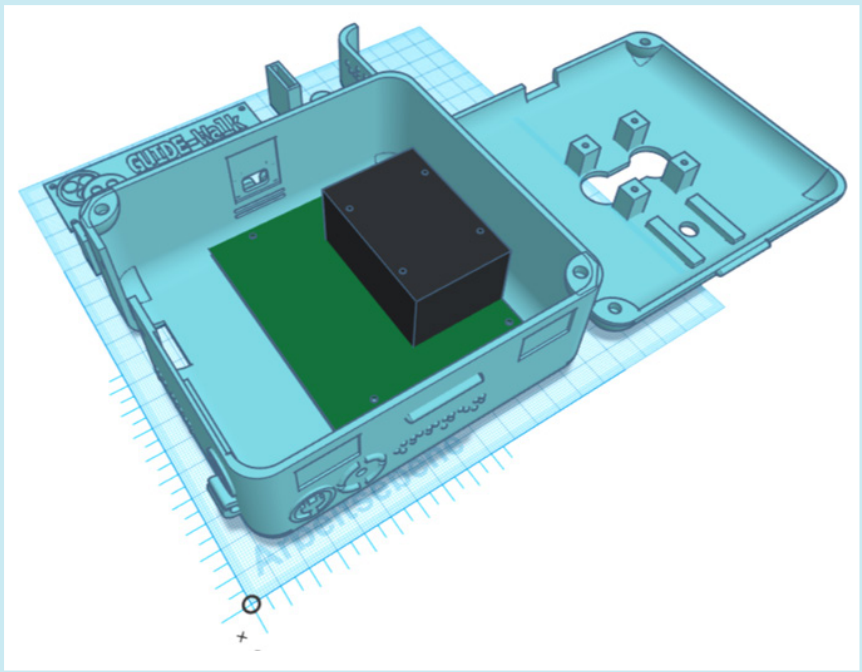

Fig. 5: Final CAD casing design

3.4 Audio Output Programming

The output of audio signals, i.e. passing on the information that has been collected and processed by the individual components to the blind wearer, is an essential element of the system. To convey warnings in a user-friendly and resource-saving way, a TTS program was used to read the warnings aloud, which was then recorded and edited for the device's program to play back. A total of 41 voicelines that can be viewed in the GitHub repo were created this way. For the distance feedback, sine waves are generated which are played back at certain time intervals. In the C++ integration, audio files must be played asynchronously, i. e. parallel to the normal program sequence. The Mixer module of the Simple DirectMedia Layer (SDL) library loads the voicelines into memory and can play them in two channels simultaneously (*audio.h*). Since the Jetson Nano™ does not have a headphone jack socket, an external sound card with a USB connection was required.

It soon became apparent, however, that the warnings generated by the program had to be filtered in some way; otherwise, they would be output continuously and randomly. To organize them, warnings are first placed in a dynamic array, called a vector. Each warning itself is an array and has the following format:

$$\{[warning\_id], [state], [cooldown], [priority]\}$$

Each audio message has an ID (defined in the header file) and a priority value from 0 to 1000. The warnings are sorted according to priority and since all have different values, it is guaranteed that, if several messages are received at the same time, always the most important one can be found. For example, the system always warns of pedestrians or traffic lights rather than notifying the presence of benches or bins, and information about time and weather is immediately interrupted when nearby objects are detected. When an element of the vector has been played, its state is marked accordingly. From then on, a cooldown counter starts, after which the entry is removed. This is important because warnings are only saved to the queue if there is no other with the same ID in the list, which prevents an accumulation of warnings of the same type. Overall, this system helps to ensure that audio messages are sorted and that the wearer is not unduly overloaded with information.

3.5 Power Management and Casing Design

Deciding on how the device was worn and designing the casing in a way that made it comfortable and suitable for everyday use as well as practical and

doi: 10.7795/320.202205

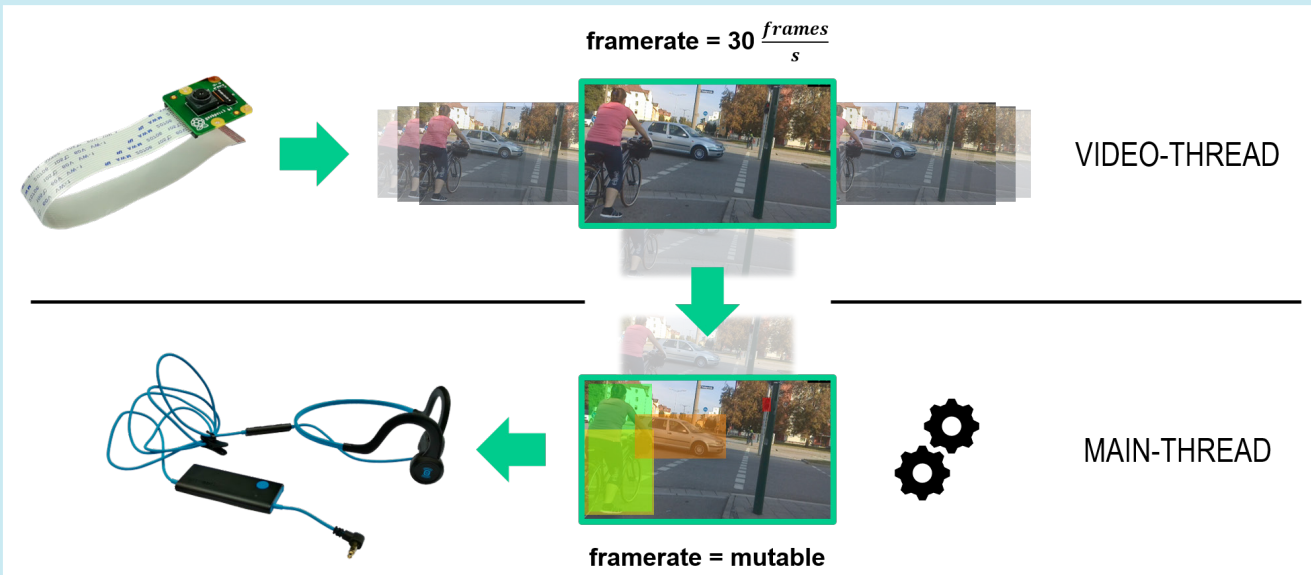

Fig. 6: Schematic illustration of the program's structure using separate threads

safe presented another challenge. After evaluating different ways of wearing the device, it proved to be best to hang the final product around the neck with a loop. The deciding factors ultimately were that this way, it can be put on and taken off quickly and easily, and the wearer is not restricted in the choice of clothing.

In contrast to the prototype, the battery has been moved to a separate box that fits in a bag or can be strapped to a belt. It can be removed using a sliding mechanism and is connected to the device via a spiral cable. The main box (see Fig. 5), which contains the Jetson Nano™ and the other components, is relatively compact with dimensions of 12.9 cm × 12 cm × 6.4 cm. It is equipped with an easily removable, but still firmly-fitting lid with a magnetic lock as well as recesses for the camera and the LiDAR sensor. On the lower half, there is a headphone socket, eyelets for the safety pins, and hooks for winding the cable. To make handling easier for blind people, all important points are marked with Braille or, if the wearer does not know Braille, with simple haptic symbols. This also applies to the container of the battery. The Jetson Nano™ is fixed in the housing with screws, which means that the device can be easily dismantled for repair purposes.

To supply the Jetson Nano™ with sufficient power in a mobile design, it was decided on providing it through the DC barrel jack. The power consumption is between 5 to 10 W on paper, but in reality, it reaches higher levels, such that the 5V⎓2.1A provided by USB are not sufficient. This has been confirmed by both self-conducted experiments and Nvidia® itself on their forums. It is most likely that the sensors and peripherals are the cause of the additional power consumed. The DC socket (5.5 mm / 2.1 mm) can deliver up to 5V⎓4A, i.e. double the current. To do this, the J48 pins must be short-circuited with a jumper. The power comes from a rechargeable battery (capacity: 8,300 mAh) that is easy to use for the blind and has a DC socket with

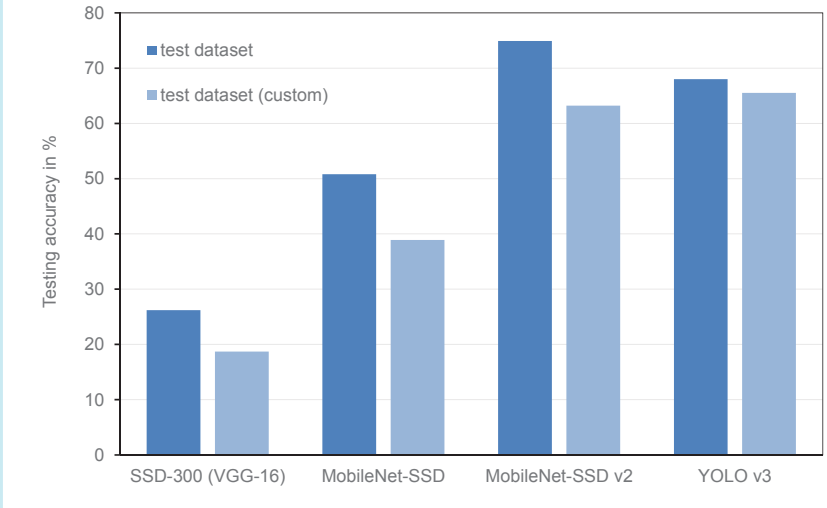

Fig. 7: Testing accuracy by dataset (mAP) compared to other state-of-the-art networks

doi: 10.7795/320.202205

an output of 12V⎓6A. To step it down, a transformer was integrated into the housing.

3.6 Optimization

After the successful implementation of the individual modules, the task was to complete the program and optimize it as far as possible. The goal hereby was to achieve the shortest possible response time and, at the same time, the lowest possible energy consumption. With a few exceptions, these optimization processes run completely in the background, are self-contained, and include AI optimizations.

In addition to reducing the model size with the training described in Section 3.1, images that are unsuitable for evaluation are sorted out in advance. This is done by calculating the Laplace variance of the camera images, whose values represent the degree of blurring. A build-in OpenCV function applies an edge-sensitive kernel to the image and the squared standard deviation of the results is calculated. If it falls below a certain threshold value, a new picture is taken. Since the image quality of the camera module suffers enormously when there is a lack of light, object recognition is deactivated at night to ensure more safety. The individual frames get reduced to a size of 64 px × 35 px, cropped to the upper third, and converted to the HSV color space. If the brightness, in this case, the mean of the "value" parameter of all pixels, is too low for too long, the AI pointer is set to a *nullpointer,* which switches off the AI. This is also communicated to the wearer.

To increase the speed of object detection, the AI calculations get carried out on the GPU, which significantly accelerates the process. The Jetson Nano™'s integrated graphics chip with Maxwell architecture was specifically designed for this purpose, but to ensure stable operation, one additionally has to define a new energy profile for the board and limit the clock frequencies of the CPU (918 MHz) and GPU (640 MHz). Otherwise, the computer would crash due to excessive energy consumption and voltage undersupply ("brownout"). The performance was throttled somewhat as a result, but this measure was necessary.

A very important tool for improving performance was multithreading. If the GStreamer pipeline delivers more images per second than the AI can process at the same time, the recordings accumulate and a time shift occurs between the just recorded frames and the ones already evaluated. To solve this fundamental problem, the camera stream is executed on a separate thread. Therefore, the thread produces a constant stream of 30 images per second, from which the main thread can copy the next image whenever necessary. Using this method (see Fig. 6), the recording rate of the camera is independent of the rest of the program.

The distance measurement and feedback also run in a different thread, so that the time interval between the signal tones is not influenced by other factors. Crashes are prevented by functions that detect errors in the sensors or other components, send a notification, and shut down the program in a controlled manner. In particular, all instances of all classes are emptied from the memory and the threads are terminated. Switching on and off is also designed to be as user-friendly as possible. By adding it to the *autostart* applications on Linux, the program runs automatically after the battery is plugged in. To shut down the device, two upward tilt gestures are required.

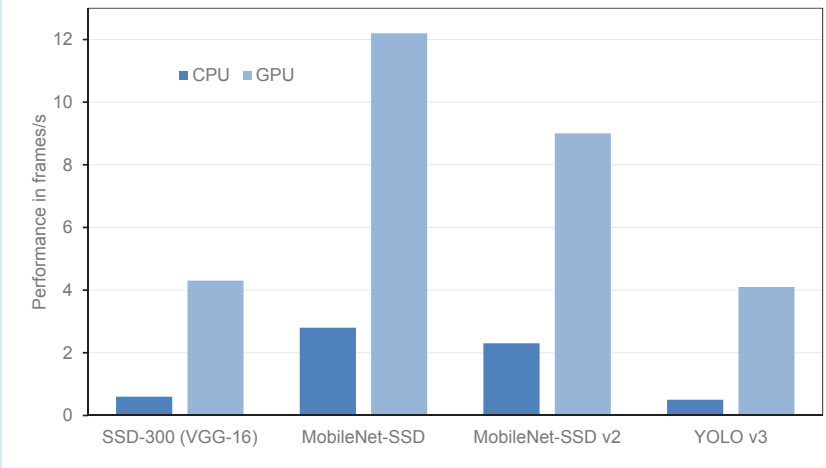

Fig. 8: Performance by object detection network models and operating mode

4. Results

4.1 Accuracy and Performance Evaluation of the AI

The final mean Average Precision (mAP) on the test dataset is 74.9 % with an average loss of 2.43. This calculation by the Caffe framework was done by taking the mAP score, determined with the precision-recall curve, over all classes and a single Intersection over Union (IoU) threshold of 0.5. The final network only takes those predictions into account where it is more than 50 % confident. In addition, a set of unrelated test images (referenced as "test dataset (custom)" in Fig. 7) was created to obtain a more meaningful value. Its entries depict the individual classes one at a time in an isolated environment. In Fig. 7, this result can be seen in

doi: 10.7795/320.202205

comparison with other similar, state-of-the-art object detection networks that were trained using the same dataset and hardware.

Further, the performance of these networks in their C++ implementation on the Jetson Nano™ was tested. A program script measured the duration of the individual inference runs per image, calculating an average value for the CPU-time per frame. If the detections were rendered on the screen, this could be interpreted as the frame rate. The achieved frames per second (FPS) for the different models depending on the operating mode are shown in Fig 8.

4.2 Accuracy Determination of the Distance Sensor

To test the accuracy, the sensor was attached at a distance of 30 cm in front of a wall, measuring the distance ten times per second for ten seconds. The same setup was repeated with a distance of 2 m and 5 m. Fig. 9 shows the change in the measured values in the course of the measurements to the actual distance.

4.3 Main Loop Execution Speed and Runtime Evaluation

When the main program was finished, the system was tested as a whole to obtain a summary statistic of its performance and see how effective the above-mentioned optimizations were. With the implementation of a simple FPS counter and the formula

$$FPS = \frac{t_{end} - t_{start}}{1{,}000{,}000\ \mu s} \tag{7}$$

the runtime of a single loop execution was measured for 100 frames. The averaged results are shown in Fig. 10. With CPU only, the program achieves an average of 0.6 FPS. Using GPU acceleration, the frame rate rises to an average of 6.5 FPS.

When it comes to determining the battery life t_{bat}, there are two possible methods. Arithmetically, it can be

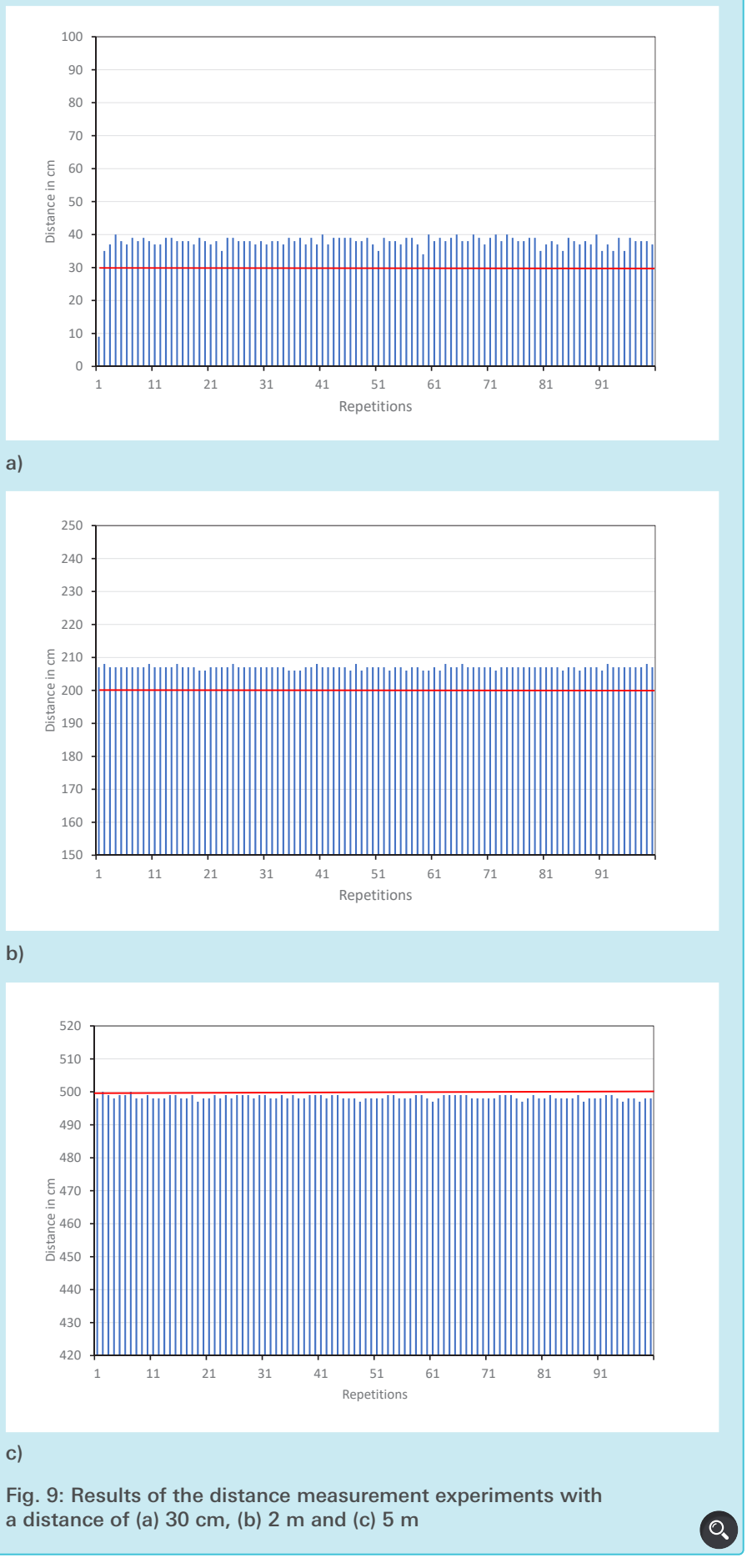

Fig. 9: Results of the distance measurement experiments with a distance of (a) 30 cm, (b) 2 m and (c) 5 m

calculated as follows:

$$t_{bat} = \frac{U \cdot Q}{P} = \frac{5\ V \cdot 8.3\ Ah}{10\ W} = 4.15\ h \quad (8)$$

An experimental determination was additionally performed to confirm the result. Using the LED lights on the rechargeable battery, it was observed at which pace it emptied, the corresponding time was measured and then extrapolated. Fig. 11 shows the results. With uninterrupted usage, the device runs for 7 to 8 hours.

5. Discussion

Of all tested networks, MobileNet-SSD v2 performs best. An accuracy percentage range of 70 to 80 % is typical for modern object detection networks. This might seem low at first, but considering that the accuracy is calculated using the predictions' deviation from the exact positions of the original bounding boxes, it is not; to our human eyes, a network is considered "accurate" even if it roughly determines the location of objects in the image, which is why the achieved mAP of 74.9 % is suitable for this application. By running a demo on some test images, a pattern of false boxes can be observed which are duplicates of larger boxes of the same object. Filtering them can lead to a further improvement of accuracy, which happens in post-processing and, therefore, cannot be measured by Caffe.

When it comes to execution time, MobileNet-SSD v2 did not achieve the best results, but since the accuracy of the fastest model is much lower, this limitation is justifiable. The frames per second were measured both with and without GPU acceleration, on average resulting in quadrupled results for the GPU application. Combining the operating speed of the AI inference and the overall program, the system achieves an average reaction time of $\frac{1}{6.5}$s = 0.15s. This is a great advantage compared to the reaction time of humans (~1s). According to experiences of the test persons in the field testings, this time is not only completely sufficient but also more "forward-looking", especially if considered that the device is only meant to be used as a supplement for the perception of the blind.

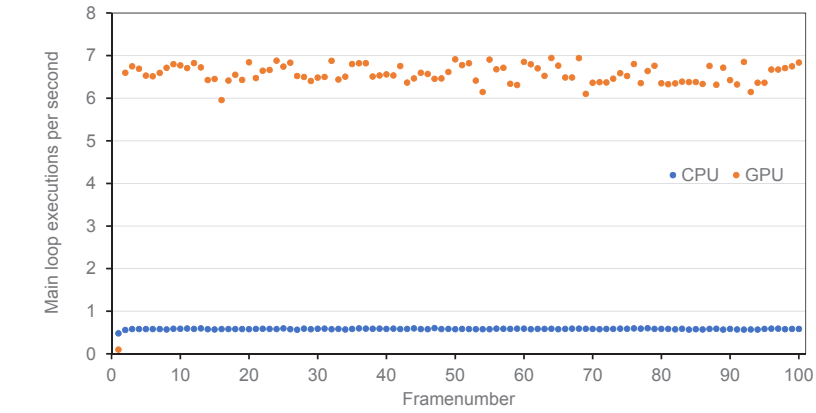

Fig. 10: Results of the runtime test

The accuracy tests of the LiDAR sensor show that, interestingly, the distance tends to be measured as too far, with this deviation from the actual value decreasing with increasing distance. At very short distances such as 30 cm, the error can be up to 10 cm, at 5 m, on the other hand, the measurements are relatively accurate. In all three experiments, the measured values show very small noise that does not affect performance. Therefore, one can assume that the sensor has a high level of accuracy since when walking, you are unlikely to reach a distance that is as small as 30 cm. The sensor readings are not susceptible to inaccuracies because of the motion during walking itself, since the time difference between the emission and the reception of the signals is in the nanosecond range.

The calculated battery life is imprecise and does not match the experimental results for several reasons: First, the power consumption of the Jetson Nano™ can only be estimated, and second, physical parameters such as losses during transformation, etc., are not included in the formula. The experimentally determined duration, however, is a solid result that adequately covers most of the time when walking outside.

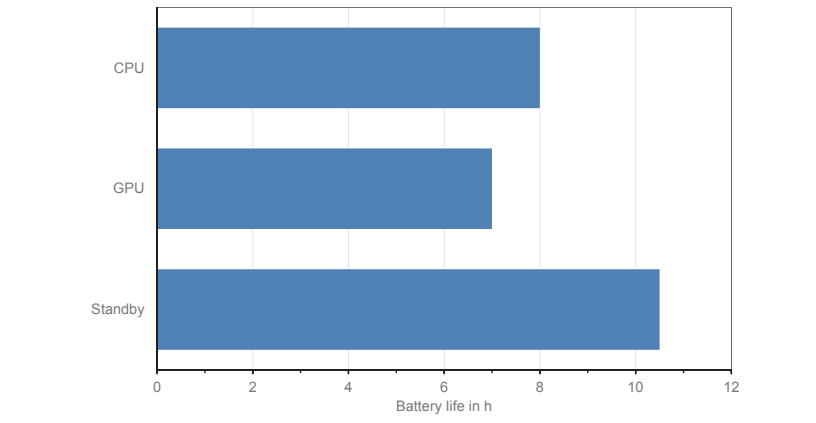

Fig. 11: Battery life (extrapolated) in different modes

doi: 10.7795/320.202205

6. Experiences of Those Affected

While developing the project, it was particularly important to obtain first-hand feedback to improve aspects that most laypeople do not recognize. For this purpose, after contacting various institutions for the blind, the device was tested by two visually impaired people in action on the street.

The first tester sees object detection with artificial intelligence as a sensible approach when it comes to recognizing unexpected or sudden events such as obstacles or red lights. He also advocates the least possible impairment of the wearer's hearing. When it comes to the operation of the device, he criticized that the system gave him too much and sometimes unreliable information. At his suggestion, the AI was improved with additional recordings and new training, and the curb detection was removed due to the lack of more functional alternatives. Overall, the guidance device can contribute to the inclusion process in his opinion.

The other tester emphasized that the assistance device can help with cuts by nearby pedestrians, such as hectic rush or (often unintentional) lack of consideration. He, therefore, sees the use of AI for obstacle and traffic light detection as suitable, but criticized the inadequate filtering and the "flood" of information that is passed on to the wearer. In line with his proposal, the system for prioritizing warnings was revised. He also complained about the lack of a spatial conception of recognized hazards, which is why the distance measurement was linked to the object recognition to only issue a warning when an obstacle is in the immediate vicinity. In summary, the insights and findings from these discussions and tests were extremely helpful and have contributed significantly to improving the device in many ways.

7. Conclusion and Prospects

Now that the composition and the functionality of the blind guide system have been explained in detail, it can be concluded that the aim of creating a navigation helper prototype for the blind has been achieved. In comparison to the previous model, the AI in particular has developed considerably and works much more precisely. Further, with the elaboration of new features and working speech output, it better represents a guide leading through traffic (although it is important to point out that the device does not replace a trained guide person). With further development, it is more than realistic that this project could make everyday life easier for the blind and visually impaired.

At the same time, the system is far from being perfect, and there are still numerous aspects waiting for improvement. As a creator of such software, one has a great responsibility towards its users. However, being open-minded and expectant about the future of guidance systems like the GUIDE-Walk is the best way to achieve further progress. In order to create a marketable product that can be offered to all those affected at a reasonable price, porting the system to a PCB with custom components is in progress. This would enable for integration into wearables, such as glasses with more compact sensors. When produced in large enough batches, the final product could be offered for a few hundred dollars, the price of a mid-range smartphone that is affordable for most. In any case, it should be stressed that AI is used for helping people and perhaps giving them a new perspective through the "eye of the computer".

Acknowledgements

Many thanks to all those who supported and helped me in planning and realizing this project! These are in particular: My parents with the good ideas, their willingness to help, and their financial support; the school of the Regensburger Domspatzen and its headmistress, Christine Lohse, without whose support my project would not have been possible; the Bavarian Youth Research Sponsor Pool and EDIsys Kft., who provided me with a budget for the realization of the project; the testers Rainer Schliermann (University of Applied Sciences (OTH) Regensburg) and Rudolf Pichlmeier (Bavarian Association of the Blind and Visually Impaired (BBSB e.V.) Oberpfalz) with their cooperation, efforts and constructive criticism; the proof-readers and professional consultants René Grünbauer and Manfred Hofmann; the head of the Institution for the Blind in Regensburg, Ulrike Weimer, with her mediation work; Moritz Walker and Christoph Högl with their advice on technical issues; Yannick Rittner, Patrick Soller, and Paul Kutzer with their spontaneous helpfulness.

Bibliography and References

[1] *„Jugend forscht" – Siegerliste 2020, p. 18.* (2020, Mar 05). Retrieved from Audi MediaCenter: https://www.audi-mediacenter.com/de/publikationen/weitere/jugend-forscht-siegerliste-2020-888

[2] José, J., Farrajota, M., Rodrigues, J., & du Buf, J. M. (2010, Jan). *A vision system for detecting paths and moving obstacles for the blind.* Retrieved from ResearchGate: https://www.researchgate.net/publication/216435222_A_vision_system_for_detecting_paths_and_moving_obstacles_for_the_blind

[3] Rodrigues, J., du Buf, J. M., & Castells, D. E. (2010, Jan). *Obstacle detection and avoidance on sidewalks.* Proc. Int. Conf. on Computer Vision-Theory and Applications (VISAPP2010), Angers, France. 2. 235-240. Retrieved from ResearchGate: https://www.researchgate.net/publication/216435190_Obstacle_detection_and_avoidance_on_sidewalks

[4] Kumar, A., & Chourasia, A. (2018, Mar). *Blind Navigation System Using Artificial Intelligence.* International Research Journal of Engineering and Technology (IRJET), vol. 5, no. 3, pp. 601-605. Retrieved from IRJET: https://www.irjet.net/archives/V5/i3/IRJET-V5I3134.pdf

[5] Xiao, A., Tong, W., Yang, L., Zeng, J., Li, Z., & Sreenath, K. (2021, Jun 28). *Robotic Guide Dog: Leading a Human with Leash-Guided Hybrid Physical Interaction.* Retrieved from arXiv: https://arxiv.org/abs/2103.14300

[6] Wachaja, A., Agarwal, P., Zink, M., Adame, M. R., Möller, K., & Burgard, W. (2017). *Navigating blind people with walking impairments using a smart walker.* Autonomous Robots, vol. 41, no. 3, pp. 555-573. Retrieved from SpringerLink: https://doi.org/10.1007/s10514-016-9595-8

[7] Ye, C., Hong, S., Qian, X., & Wu, W. (2016). *Co-Robotic Cane: A New Robotic Navigation Aid for the Visually Impaired.* IEEE Systems, Man, and Cybernetics Magazine, vol. 2, no. 2, pp. 33-42. Retrieved from IEEE Xplore: https://ieeexplore.ieee.org/document/7549220

[8] *Seeing AI™* on Microsoft: https://www.microsoft.com/en-us/ai/seeing-ai

[9] *OrCam® MyEye™ 2.0* on OrCam: https://www.orcam.com/en/myeye2/

[10] *InnoMake™* on Tec-Innovation: https://www.tec-innovation.com/en/innomake-en/

[11] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016, Dec 29). *SSD: Single Shot MultiBox Detector. p. 4, fig. 2.* Retrieved from arXiv: https://arxiv.org/abs/1512.02325

[12] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., . . . Adam, H. (2017, Apr 17). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, p. 6, table 8.* Retrieved from arXiv: https://arxiv.org/abs/1704.04861

[13] *MobileNetV2-SSD* on GitHub: https://github.com/zhanghanbin3159/MobileNetV2-SSD

[14] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., . . . Zitnick, C. L. (2014). *Microsoft COCO: Common Objects in Context.* Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8693. Retrieved from SpringerLink: https://doi.org/10.1007/978-3-319-10602-1_48

[15] Everingham, M., Van Gool, L., K. I. Williams, C., Winn, J., & Zisserman, A. (2010). *The PASCAL Visual Object Classes (VOC) Challenge.* Int J Comput Vis 88, 303–338. Retrieved from SpringerLink: https://doi.org/10.1007/s11263-009-0275-4

[16] *Ampel-Pilot-Dataset* on GitHub: https://github.com/patVInta/Ampel-Pilot-Dataset

[17] Blanco, J.-L. (2013, May 09). *A tutorial on SE(3) transformation parameterizations and on-manifold optimization.* Retrieved from CiteSeerX: https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.468.5407

[18] International Organization for Standardization. (1975). *Standard Atmosphere.* ISO 2533:1975.

[19] Weisstein, E. W. *Standard Deviation.* Retrieved from MathWorld--A Wolfram Web Resource: https://mathworld.wolfram.com/StandardDeviation.html

doi: 10.7795/320.202205

JUNGE wissenschaft
Verlag: Physikalisch-Technische Bundesanstalt PTB

Publiziere auch Du hier!

Forschungsarbeiten von Schüler/Inne/n und Student/Inn/en

In der Jungen Wissenschaft werden Forschungsarbeiten von SchülerInnen, die selbstständig, z. B. in einer Schule oder einem Schülerforschungszentrum, durchgeführt wurden, veröffentlicht. Die Arbeiten können auf Deutsch oder Englisch geschrieben sein.

Wer kann einreichen?

SchülerInnen, AbiturientInnen und Studierende ohne Abschluss, die nicht älter als 23 Jahre sind.

Was musst Du beim Einreichen beachten?

Lies die Richtlinien für Beiträge. Sie enthalten Hinweise, wie Deine Arbeit aufgebaut sein soll, wie lang sie sein darf, wie die Bilder einzureichen sind und welche weiteren Informationen wir benötigen. Solltest Du Fragen haben, dann wende Dich gern schon vor dem Einreichen an die Chefredakteurin Sabine Walter.

Lade die Erstveröffentlichungserklärung herunter, drucke und fülle sie aus und unterschreibe sie.

Dann sende Deine Arbeit und die Erstveröffentlichungserklärung per Post an:

Chefredaktion Junge Wissenschaft
Dr.-Ing. Sabine Walter
Paul-Ducros-Straße 7
30952 Ronnenberg
Tel: 05109 / 561508
Mail: sabine.walter@verlag-jungewissenschaft.de

Wie geht es nach dem Einreichen weiter?

Die Chefredakteurin sucht einen geeigneten Fachgutachter, der die inhaltliche Richtigkeit der eingereichten Arbeit überprüft und eine Empfehlung ausspricht, ob sie veröffentlicht werden kann (Peer-Review-Verfahren). Das Gutachten wird den Euch, den AutorInnen zugeschickt und Du erhältst gegebenenfalls die Möglichkeit, Hinweise des Fachgutachters einzuarbeiten.

Die Erfahrung zeigt, dass Arbeiten, die z. B. im Rahmen eines Wettbewerbs wie **Jugend forscht** die Endrunde erreicht haben, die besten Chancen haben, dieses Peer-Review-Verfahren zu bestehen.

Schließlich kommt die Arbeit in die Redaktion, wird für das Layout vorbereitet und als Open-Access-Beitrag veröffentlicht.

Was ist Dein Benefit?

Deine Forschungsarbeit ist nun in einer Gutachterzeitschrift (Peer-Review-Journal) veröffentlicht worden, d. h. Du kannst die Veröffentlichung in Deine wissenschaftliche Literaturliste aufnehmen. Deine Arbeit erhält als Open-Access-Veröffentlichung einen DOI (Data Object Identifier) und kann von entsprechenden Suchmaschinen (z. B. BASE) gefunden werden.

Die Junge Wissenschaft wird zusätzlich in wissenschaftlichen Datenbanken gelistet, d. h. Deine Arbeit kann von Experten gefunden und sogar zitiert werden. Die Junge Wissenschaft wird Dich durch den Gesamtprozess des Erstellens einer wissenschaftlichen Arbeit begleiten – als gute Vorbereitung auf das, was Du im Studium benötigst.

doi: 10.7795/320.202205

Richtlinien für Beiträge

Für die meisten Autor/Inn/en ist dies die erste wissenschaftliche Veröffentlichung. Die Einhaltung der folgenden Richtlinien hilft allen – den Autor/innen/en und dem Redaktionsteam

Die Junge Wissenschaft veröffentlicht Originalbeiträge junger AutorInnen bis zum Alter von 23 Jahren.

- Die Beiträge können auf Deutsch oder Englisch verfasst sein und sollten nicht länger als 15 Seiten mit je 35 Zeilen sein. Hierbei sind Bilder, Grafiken und Tabellen mitgezählt. Anhänge werden nicht veröffentlicht. Deckblatt und Inhaltsverzeichnis zählen nicht mit.

- Formulieren Sie eine eingängige Überschrift, um bei der Leserschaft Interesse für Ihre Arbeit zu wecken, sowie eine wissenschaftliche Überschrift.

- Formulieren Sie eine kurze, leicht verständliche Zusammenfassung (maximal 400 Zeichen).

- Die Beiträge sollen in der üblichen Form gegliedert sein, d. h. Einleitung, Erläuterungen zur Durchführung der Arbeit sowie evtl. Überwindung von Schwierigkeiten, Ergebnisse, Schlussfolgerungen, Diskussion, Liste der zitierten Literatur. In der Einleitung sollte die Idee zu der Arbeit beschrieben und die Aufgabenstellung definiert werden. Außerdem sollte sie eine kurze Darstellung schon bekannter, ähnlicher Lösungsversuche enthalten (Stand der Literatur). Am Schluss des Beitrages kann ein Dank an Förderer der Arbeit, z. B. Lehrer und Sponsoren, mit vollständigem Namen angefügt werden. Für die Leser kann ein Glossar mit den wichtigsten Fachausdrücken hilfreich sein.

- Bitte reichen Sie alle Bilder, Grafiken und Tabellen nummeriert und zusätzlich als eigene Dateien ein. Bitte geben Sie bei nicht selbst erstellten Bildern, Tabellen, Zeichnungen, Grafiken etc. die genauen und korrekten Quellenangaben an (siehe auch Erstveröffentlichungserklärung). Senden Sie Ihre Bilder als Originaldateien oder mit einer Auflösung von mindestens 300 dpi bei einer Größe von 10 · 15 cm! Bei Grafiken, die mit Excel erstellt wurden, reichen Sie bitte ebenfalls die Originaldatei mit ein.

- Vermeiden Sie aufwendige und lange Zahlentabellen.

- Formelzeichen nach DIN, ggf. IUPAC oder IUPAP verwenden. Gleichungen sind stets als Größengleichungen zu schreiben.

- Die Literaturliste steht am Ende der Arbeit. Alle Stellen erhalten eine Nummer und werden in eckigen Klammern zitiert (Beispiel: Wie in [12] dargestellt …). Fußnoten sieht das Layout nicht vor.

- Reichen Sie Ihren Beitrag sowohl in ausgedruckter Form als auch als PDF ein. Für die weitere Bearbeitung und die Umsetzung in das Layout der Jungen Wissenschaft ist ein Word-Dokument mit möglichst wenig Formatierung erforderlich. (Sollte dies Schwierigkeiten bereiten, setzen Sie sich bitte mit uns in Verbindung, damit wir gemeinsam eine Lösung finden können.)

- Senden Sie mit dem Beitrag die Erstveröffentlichungserklärung ein. Diese beinhaltet im Wesentlichen, dass der Beitrag von dem/der angegebenen AutorIn stammt, keine Rechte Dritter verletzt werden und noch nicht an anderer Stelle veröffentlicht wurde (außer im Zusammenhang mit **Jugend forscht** oder einem vergleichbaren Wettbewerb). Ebenfalls ist zu versichern, dass alle von Ihnen verwendeten Bilder, Tabellen, Zeichnungen, Grafiken etc. von Ihnen veröffentlicht werden dürfen, also keine Rechte Dritter durch die Verwendung und Veröffentlichung verletzt werden. Entsprechendes Formular ist von der Homepage www.junge-wissenschaft.ptb.de herunterzuladen, auszudrucken, auszufüllen und dem gedruckten Beitrag unterschrieben beizulegen.

- Schließlich sind die genauen Anschriften der AutorInnen mit Telefonnummer und E-Mail-Adresse sowie Geburtsdaten und Fotografien (Auflösung 300 dpi bei einer Bildgröße von mindestens 10 · 15 cm) erforderlich.

- Neulingen im Publizieren werden als Vorbilder andere Publikationen, z. B. hier in der Jungen Wissenschaft, empfohlen.

doi: 10.7795/320.202205

Impressum

Junge Wissenschaft
c/o Physikalisch-Technische
Bundesanstalt (PTB)
www.junge-wissenschaft.ptb.de

Redaktion
Dr. Sabine Walter, Chefredaktion
Junge Wissenschaft
Paul-Ducros-Str. 7
30952 Ronnenberg
E-Mail: sabine.walter@verlag-jungewissenschaft.de
Tel.: 05109 / 561 508

Verlag
Dr. Dr. Jens Simon,
Pressesprecher der PTB
Bundesallee 100
38116 Braunschweig
E-Mail: jens.simon@ptb.de
Tel.: 0531 / 592 3006
(Sekretariat der PTB-Pressestelle)

Design & Satz
Sebastian Baumeister
STILSICHER - Grafik & Werbung
E-Mail: baumeister@stilsicher.design
Tel.: 05142 / 98 77 89