International Journal of Research in Engineering, Science and Management
Volume 7, Issue 4, April 2024
https://www.ijresm.com | ISSN (Online): 2581-5792

Step Simple – Guiding the Visually Challenged

Karthik Ganesh[1*], Advaith Prasad[2], Mohammed[3], Ashwitha A. Shetty[4]
[1,2,3]*Student, Department of Computer Science and Engineering, A. J. Institute of Engineering and Technology, Mangalore, India*
[4]*Assistant Professor, Department of Computer Science and Engineering, A. J. Institute of Engineering and Technology, Mangalore, India*

***Abstract*: This project presents the design and implementation of a smart blind stick prototype aimed at enhancing the mobility and safety of visually impaired individuals. The blind stick integrates an ultrasonic sensor, USB web camera, speakers, to provide object detection. The prototype leverages a Raspberry pi controller to efficiently manage the sensor data and interactions.**

***Keywords*: embedded system, raspberry pi, ultrasonic sensor, YOLO.**

1. Introduction

The project introduces a groundbreaking initiative focusing on the design and implementation of a smart belt prototype named the Blind Stick. This innovative device is tailored to augment the mobility and safety of visually impaired individuals by integrating cutting-edge technologies. At its core, the Blind Stick utilizes a Raspberry pi microcontroller to efficiently manage various components, including an ultrasonic sensor, USB web camera, speakers. The combination of these elements enables the prototype to offer essential functionalities such as object detection, alert generation, and location sharing.

The ultrasonic sensor plays a pivotal role in enhancing user safety by detecting obstacles in the path of the visually impaired individual.

The Blind Stick prototype takes a significant leap forward by incorporating computer vision techniques, leveraging the YOLO (You Only Look Once) framework. Through the use of a USB web camera connected to a PC, the Blind Stick captures images, and YOLO identifies detected objects. The user is then provided with audio alerts through earphones, offering a more comprehensive understanding of their surroundings.

2. Embedded System Implementation

A. Introduction

An embedded system is one kind of a computer system mainly designed to perform several tasks like to access, process, store and also control the data in various electronics-based systems. Embedded systems are a combination of hardware and software where software is usually known as firmware that is embedded into the hardware. One of its most important characteristics of these systems is, it gives the o/p within the time limits. Embedded systems support to make the work more perfect and convenient. So, we frequently use embedded systems in simple and complex devices too. The applications of embedded systems mainly involve in our real life for several devices like microwave, calculators, etc.

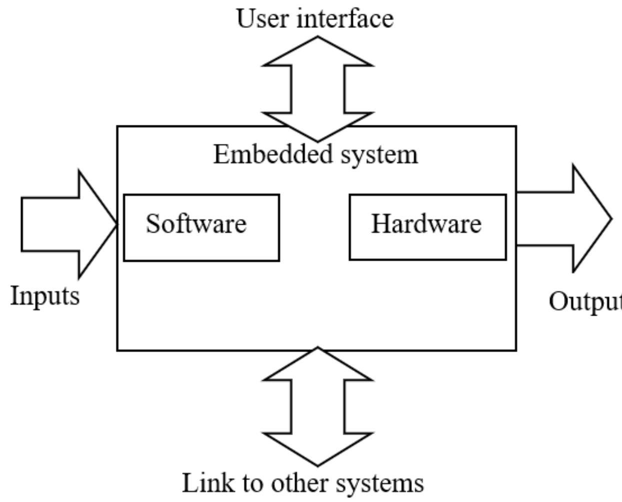

Fig. 1. Overview of embedded system

B. Embedded System Hardware

As with any electronic system, an embedded system requires a hardware platform on which it performs the operation. Embedded system hardware is built with a microprocessor or microcontroller. The embedded system hardware has elements like input output (I/O) interfaces, user interface, memory and the display. Usually, an embedded system consists of:

- Power Supply
- Processor
- Memory
- Timers
- Serial communication ports

Embedded systems use different processors for its desired operation. Some of the processors used are

- Microprocessor
- Microcontroller
- Digital signal processor

C. Embedded System Software

The embedded system software is written to perform a specific function. It is typically written in a high-level format and then compiled down to provide code that can be lodged within a non-volatile memory within the hardware. An embedded system software is designed to keep in view of the

*Corresponding author: karthikganesh0104@gmail.com

three limits:

- Availability of system memory
- Availability of processor's speed
- When the system runs continuously, there is a need to limit power dissipation for events like stop, run and wake up.

3. Methodology

A. Flow Diagram

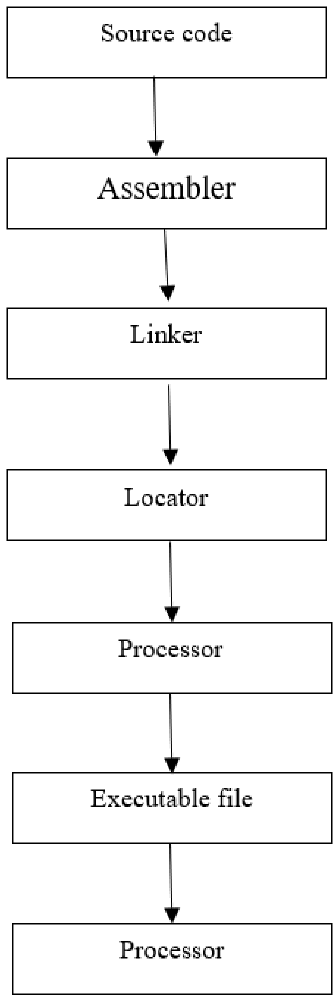

Fig. 2. Flow diagram

The flowchart in Figure above outlines the Flow of burning source code to processor:

1. *Source Code:* This is where the process begins. Source code is the human-readable form of a program written in a programming language like C, C++, Java, etc.
2. *Assembler:* The source code is first passed through an assembler. The assembler converts the source code into machine code, which is a low-level representation of the program that can be understood by the computer's processor. This machine code is often referred to as object code.
3. *Linker:* The object code produced by the assembler may have references to functions or variables that are defined in other parts of the program or in external libraries. The linker is responsible for resolving these references by linking together various object files and libraries to create a single executable file.
4. *Locator:* In some systems, after linking, there might still be unresolved memory addresses or relocation information in the executable. The locator is a hypothetical step representing the process of resolving these addresses and preparing the executable for loading into memory.
5. *Executable File:* The output of the linker is an executable file. This file contains the machine code for the program, along with any necessary data and metadata, in a format that the operating system can understand.
6. *Processor:* Finally, the executable file is loaded into memory and executed by the processor. The processor interprets the instructions in the executable file and performs the necessary computations to run the program.

Each step in this process is crucial for converting the human-readable source code into instructions that the processor can execute. This diagram illustrates the flow of data and transformations as the program moves through each stage of compilation and execution.

4. Conclusion

In conclusion, the development and implementation of the Blind Stick smart belt prototype represent a significant leap forward in addressing the challenges faced by visually impaired individuals in terms of mobility and safety. By integrating advanced technologies such as ultrasonic sensors, a USB web camera, speakers, and a switch, the Blind Stick offers a multifaceted solution to enhance the independence and security of its users. The ultrasonic sensor plays a crucial role in detecting obstacles, providing real-time alerts through a buzzer to warn users of potential collisions. The incorporation of computer vision techniques, specifically the YOLO framework, adds an extra layer of functionality to the Blind Stick. By capturing and analyzing images from a USB web camera, the prototype can identify objects in the user's surroundings and convey this information through audio alerts, delivered via earphones. This not only improves object recognition but also contributes to a more comprehensive understanding of the environment. The collective integration of sensors and modules in the Blind Stick prototype underscores its potential to make a substantial impact on the lives of visually impaired individuals. The obstacle detection, emergency alert system, challenges faced by the visually impaired community. As with any innovative project, there is room for future enhancements. Potential areas of improvement include refining the prototype for increased usability, optimizing the object detection algorithm for greater accuracy.

References

[1] J. Borenstein and Y. Koren, "Error eliminating rapid ultrasonic firing for mobile robot obstacle avoidance," IEEE Transactions on Robotics and Automation, 1985.

[2] S. Durgadevi, K. Thirupurasundari, C. Komathi, and S. Mithun Balaji, "Smart machine learning system for blind assistance," in International Conference on Power, Energy, Control and Transmission Systems, 2020.
[3] Xilin Chen, Jie Yang, Jing Zhang, and Alex Waibel, "Automatic detection and recognition of signs from natural scenes," IEEE Transactions on Image Processing, vol. 13, no. 1, January 2004.
[4] Sunit Vaidya, Niti Shah, Naisha Shah, and Prof. Radha Shankarmani, "Real-time object detection for visually challenged people," in International Conference on Intelligent Computing and Control Systems (ICICCS 2020), 2020.
[5] Amit Ghosh, Shamsul Arefeen Al Mahmud, Thajid Ibna Rouf Uday, and Dewan Md. Farid, "Assistive technology for visually impaired using TensorFlow object detection in Raspberry Pi," in 2020 IEEE Region 10 Symposium (TENSYMP), 5-7 June 2020.
[6] Megha P. Arakeri, Keerthana N. S., Madhura M., Anusha Sankar, and Tazeen Munnavar, "Assistive technology for the visually impaired using computer vision," in International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018.
[7] Sunil Kumar, Dibya Nandan Mishra, Shahid Mohammad Ganie, R. Bharathikannan, and K. Vijayakanthan, " Artificial Intelligence Solutions for the Visually Impaired: A Review," 2023.
[8] Jiaji Wang, Shuihua Wang, Yudong Zhang, "Artificial intelligence for visually impaired," in Displays, Volume 77, April 2023.
[9] Risheng Liu, Zhu Liu, Jinyuan Liu, Xin Fan, and Zhongxuan Luo, "A task-guided, implicitly-searched and meta initialized deep model for image fusion," in Computer Vision and Pattern Recognition, vol. 14, no. 8, August 2015.
[10] Yuanfeng Ji, Zhe Chen, Enze Xie, Lanqing Hong, Xihui Liu, Zhaoqiang Liu, Tong Lu, Zhenguo Li, Ping Luo, "DDP: Diffusion Model for Dense Visual Prediction," 2023 IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2023, pp. 21684-21695.