PATHPAL

Re-imagining Mobility for the Visually Impaired

Inventors

- *Bhavesh Suryanarayanan*
- *Kartik Patil*
- *Sripathiravi*
- *Mayank Sharma*
- *Rohit K*
- *Khyati*
- *Yagnik*
- *Asmitha*
- *Mohini*
- *Akshitha*

22 - 01 - 2025

Sahaay, IITM

Contents

1. Abstract

Pathpal is an innovative assistive device designed to enhance the mobility and independence of visually impaired individuals. The vision module is equipped with a camera module that leverages advanced CNN architectures and image processing algorithms to detect obstacles, find groceries and provide road navigation assistance in real time.

The Gripper module is an attachment to the four segmented foldable stick. It comes with a user-friendly 3x3 grid of coin vibrators which provide quick haptic feedback to the user. These vibrators provide a tactile sense of direction by activating specific vibrators corresponding to the detected object's location. The Gripper module is controlled by an ESP32-based control system with a compact custom PCB.

Designed with cost-efficiency and ease of use in mind, PathPal combines cutting-edge technology with practical functionality to deliver a transformative solution in assistive technology.

2. Background

Recent data shows that approximately **43 million** people worldwide are blind, with around **9 million** in India alone. Visually impaired individuals commonly use a white cane to navigate their surroundings by detecting obstacles through touch. Along with a traditional white cane, they often require assistance from other people for daily activities like crossing the road or grocery shopping.

3. Problem Statement

While the white cane has been a valuable tool for visually impaired individuals, it falls short in detecting moving vehicles, protruding obstacles, overhanging barriers and hollow objects, putting their safety at risk.

There is still a need for a technology which enables visually impaired people to completely be independent and carry out simple daily activities like grocery picking with negligible assistance. This is where PathPal will make an impact.

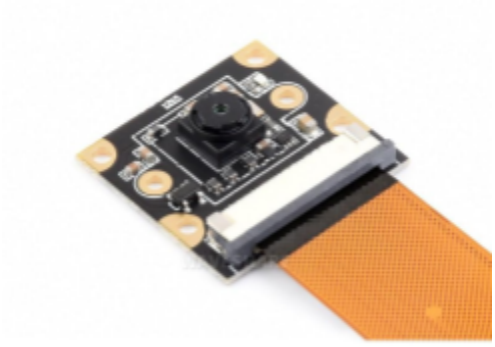
4.Technical Description

4.1.Vision Module

The Vision module of pathpal consists of the following

1. Camera Module
2. Microprocessor (Jetson Nano)
3. Microphone

4.1.1. Camera Module



Waveshare IMX-219
Camera Module

Waveshare's **IMX-219** provides 8-Megapixel resolution, providing clear and detailed images essential for accurate object detection and obstacle recognition. The 77° wide-angle lens covers a broader field of view, enabling the camera to capture more of the user's surroundings in a single frame.The IMX219 module is highly compatible with embedded platforms like Jetson

Nano. CSI (Camera Serial Interface) of the module can transmit large amounts of video data quickly. The IMX219 supports video capture at up to 30 frames per second (FPS) at full resolution, ensuring smooth real-time image processing. It consumes less power and is very compact making it suitable for this application.

4.1.2. Jetson Nano

NVDIA Jetson Nano Devkit

The **NVIDIA Jetson Nano** platform serves as the core computational unit for PathPal's vision module, providing the advanced processing power necessary for real-time computer vision tasks. Designed specifically for edge AI applications, Jetson Nano enables PathPal to deliver high-performance with low-latency.

- With up to **472 GFLOPS (Giga Floating Point Operations Per Second)** of AI performance, the Jetson Nano can efficiently run neural networks such as YOLOv8 for object detection and MiDaS for depth mapping
- Optimized for low power consumption (as low as 5W), the Jetson Nano is perfect for battery-powered devices
- Its small size and lightweight design allow easy integration into PathPal
- The Jetson Nano supports the MIPI CSI-2 interface, enabling seamless integration with the Waveshare IMX219 camera module.

The Jetson Nano is also interfaced with a microphone to obtain voice input from the user

Chest belt

A chest belt supports the vision module. The camera and jetson are placed in the centre so that maximum field of view is captured by the camera without any obstruction. The camera captures the front view of the user from top to bottom.

4.2. Gripper Module

4.2.1. The Design

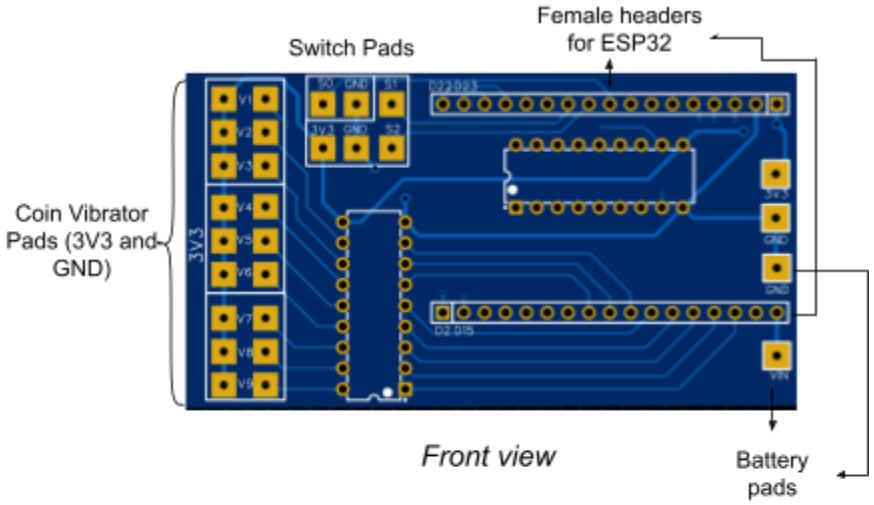
The 3d printed module consists of a 6mm thick cylindrical surface for the user to hold. This front face of the surface consists of a 3x3 grid of circular holes through which coin vibrators are placed. The user must position his/her fingers over this grid. Below the vibrators is a 3-state slide switch which allows the user to switch between the modes - Obstacle detection, grocery picking and road navigation.

The lower back of the model consists of a simple on-off switch. Users can turn off the cane when not in use to conserve power. The gripper module is attached onto a 4 segmented foldable white cane. This cane is fixed on to the lower part of the gripper module which can be detached from the upper portion by twisting. This provides access to the electronics to update code or sort loose connections.

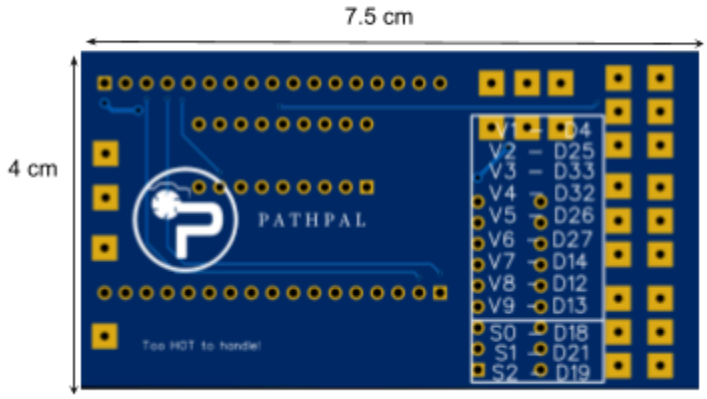
4.2.2. Electronics

The 3x3 grid of **Coin vibrators** provide haptic feedback to the user. They are disc-shaped vibration motors. Their compact size, low power consumption, and precise vibration output make them ideal for conveying tactile information. The model also consists of two switches- 1) slide switch to switch between various modes, 2) An ON-OFF switch.

These devices are controlled by a custom designed **Printed Circuit Board** placed in the cane.



Front view



Back view

An esp32 is mounted on the PCB through female header pins. The coin vibrators are controlled by the GPIO pins of the ESP32. The coin vibrators have a current requirement of 50-60mA. So, the GPIO pins are connected to a Darlington transistor array IC which amplifies the current. The outputs of the transistor array are connected to the coin vibrators. The other terminal of the coin

vibrator is connected to Ground. These connections are made by soldering the vibrators to the pads on the PCB.

There is a separate set of GPIO pads for the slide switch. There is a VIN pad through which the PCB draws power from the battery. Some extra pads are also provided for potential usage.

The Gripper module is powered by a **18650 Lithium-ion Battery.** The battery is placed in a battery holder which is connected to the TP4056 module. This module provides a c-type port, thus allowing the user to recharge the battery with his/her phone charger. The charger module is then connected to the VIN and GND pins of the PCB, powering the ESP32 and other components. An ON-OFF switch is also connected to the battery which disconnects it from the circuit, conserving power when not in use.

The ESP32 in the gripper module is connected to the Jetson Nano of the Vision module wirelessly via Wi-Fi. The ESP32 informs the Nano regarding the mode desired by the user, while the nano sends the coin vibrator states to the ESP32.

4.3.Software

4.3.1. Obstacle Detection

4.3.1.1. Depth Mapping

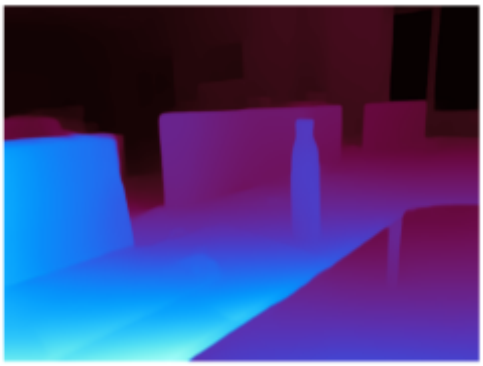
Depth mapping is a critical component of the vision module in Pathpal allowing the device to accurately assess the spatial relationships between the user and their environment.

Unlike traditional depth mapping methods like stereo cam or LIDAR, we generate the Depth map using a monocular cam using a deep learning model - **MiDaS (Monocular Depth Estimation)**.

Midas is an open source model train on large scale indoor and outdoor datasets allowing it to generate accurate depth information even from challenging real world scenes. This eliminates the need for additional sensors.

Here is a colour map generated by MiDaS. Closer are brighter than the farther ones.

Image



Depth Map

Distance Anchoring

MiDaS generates a relative depth map of the surroundings but does not provide absolute distance measurements. To obtain the absolute distance of obstacles, we employ a technique called **distance anchoring**. In this approach, a small object is placed at a known distance in front of the camera. From the generated depth map, we identify the pixel corresponding to this object. Since the actual distance to this reference object is known, we use it as a calibration point and linearly interpolate the distances for all other pixels in the depth map. This method effectively converts the relative depth values into meaningful absolute distances, enabling accurate obstacle detection and navigation.

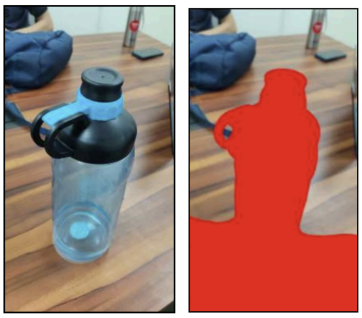
Performance

MiDaS gives an accurate depth representation of the surroundings. Being *Pytorch* based, It also enables GPU execution making it suitable for real timer purposes. We were able to run the most sophisticated MiDaS model (MiDaS Large) with a latency of just **110 ms**

Thresholding

The motive of generating a Depth map is to identify the obstacles in front of the user that are close enough to cause harm. Thus we set a distance threshold of **1m.** Any object captured in

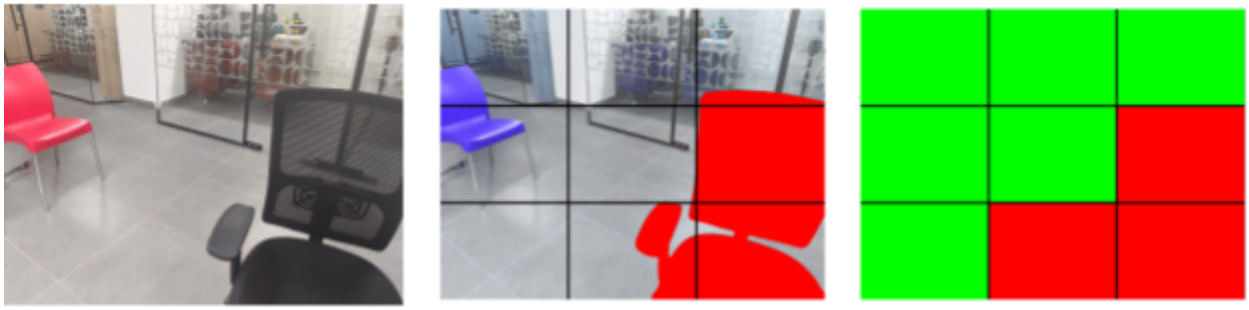
the depth map which has a distance value less than the threshold will be penalized as an obstacle. In the below picture the pixels which are closer than the threshold are masked in red colour



Original image *Mask of region nearer than threshold*

Mapping To Coin Vibrators

A 3x3 grid of rectangles is drawn over the selected region of an image and the presence of an obstacle is checked within each rectangle. For an obstacle to be present in a rectangle, we fix a threshold that at least 20% of the pixels with it must be marked as an obstacle. This is done for providing a more precise sense of direction. For each rectangle marked as an obstacle, the corresponding coin vibrator is activated.



Captured Image *Masked Image* *Coin Vibrator matrix*

Red indicates activated and green indicates deactivated

4.3.1.2. Road Segmentation

One challenge in using depth maps to identify obstacles is the unintended recognition of the road as an obstacle. Since the depth map captures the road surface in every frame, the portion

of the road close to the user is sometimes incorrectly identified as an obstacle. To resolve this, a clear distinction between the road and obstacles must be achieved. This involves accurately segmenting the road from the surrounding obstacles, ensuring the system focuses only on actual hazards while ignoring the road surface.

The pixels corresponding to the road are identified using **semantic segmentation.** The three popular architectures for segmentation are - *Mask-RCNN*, *Segnet* and *U-Net*.

We chose to use U-Net and here's why:

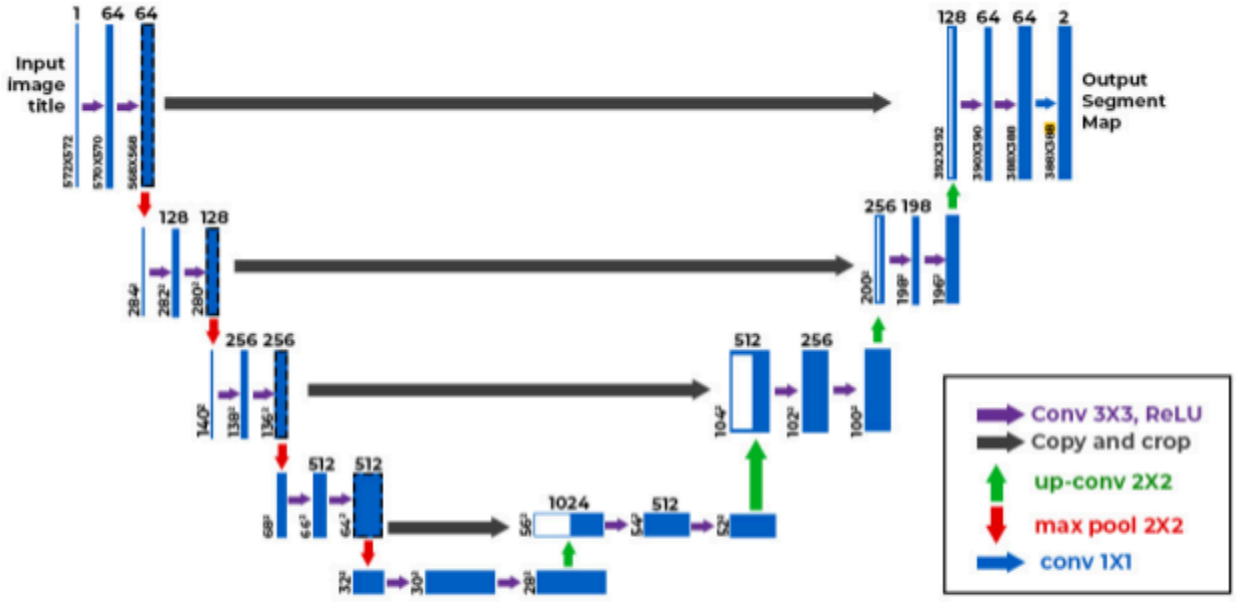
Mask R-CNN (Mask Region-Based Convolutional Network):

Mask R-CNN performs instance segmentation, meaning it can distinguish between multiple instances of the same object. However, this level of complexity is unnecessary for our use case, as we do not need to differentiate between multiple roads in the image. For our application, simpler semantic segmentation models, which classify each pixel as belonging to a specific class (e.g., road or obstacle) without distinguishing between instances, are more efficient and sufficient. Thus, using Mask R-CNN would be overkill.

As we were left with Segnet and U-Net, we tried them both and U-Net performed better due to the following reasons

- Road images are complex due to varying conditions like lighting, trees, signs, potholes, road intersections, etc. U-Net being equipped with an encoding-decoder architecture with skip connections, could perform better and gave a higher accuracy.
- Unet can be trained with a smaller dataset
- We also found that the U-net had faster runtime than Segnet which is crucial for the real-time nature of the project

U-Net Architecture



U-Net Architecture

U-Net is a deep learning architecture specifically designed for semantic segmentation tasks, excelling in scenarios requiring pixel-level precision. Originally developed for biomedical image segmentation, U-Net is characterized by its encoder-decoder structure with **symmetric skip connections**, enabling it to capture both high-level and fine-grained features.

1. **Encoder (Contracting Path)**

The encoder is responsible for extracting **hierarchical features** from the input image. It consists of several stages, each comprising:

- **Two 3x3 Convolution Layers**: Each convolution layer is followed by a ReLU activation function to introduce non-linearity.
- **Batch Normalization** : To stabilize training and accelerate convergence **Downsampling via 2x2 Max Pooling**: Reduces the spatial dimensions by half at each stage while increasing the number of feature channels.

2. Decoder (Expanding Path)

The decoder reconstructs the segmentation map by progressively upsampling the feature maps while integrating spatial details. Each stage includes:

- **Upsampling (Transposed Convolutions or Interpolation)**: Increases the spatial dimensions of the feature maps to restore the original resolution.
- **Skip Connections**: Features from the encoder are concatenated with the upsampled features. These connections allow the decoder to leverage both high-level semantic information (from deep layers) and low-level spatial details (from shallow layers).
- **Two 3x3 Convolution Layers**: Applied after concatenation, followed by ReLU activation to refine the feature maps

Performance

The model was built and trained with pytorch. It performed well on the test data with a Dice **coefficient** of **96%.** When the model was run in real time with GPU acceleration, it generated the masks with a latency of just **90ms.**

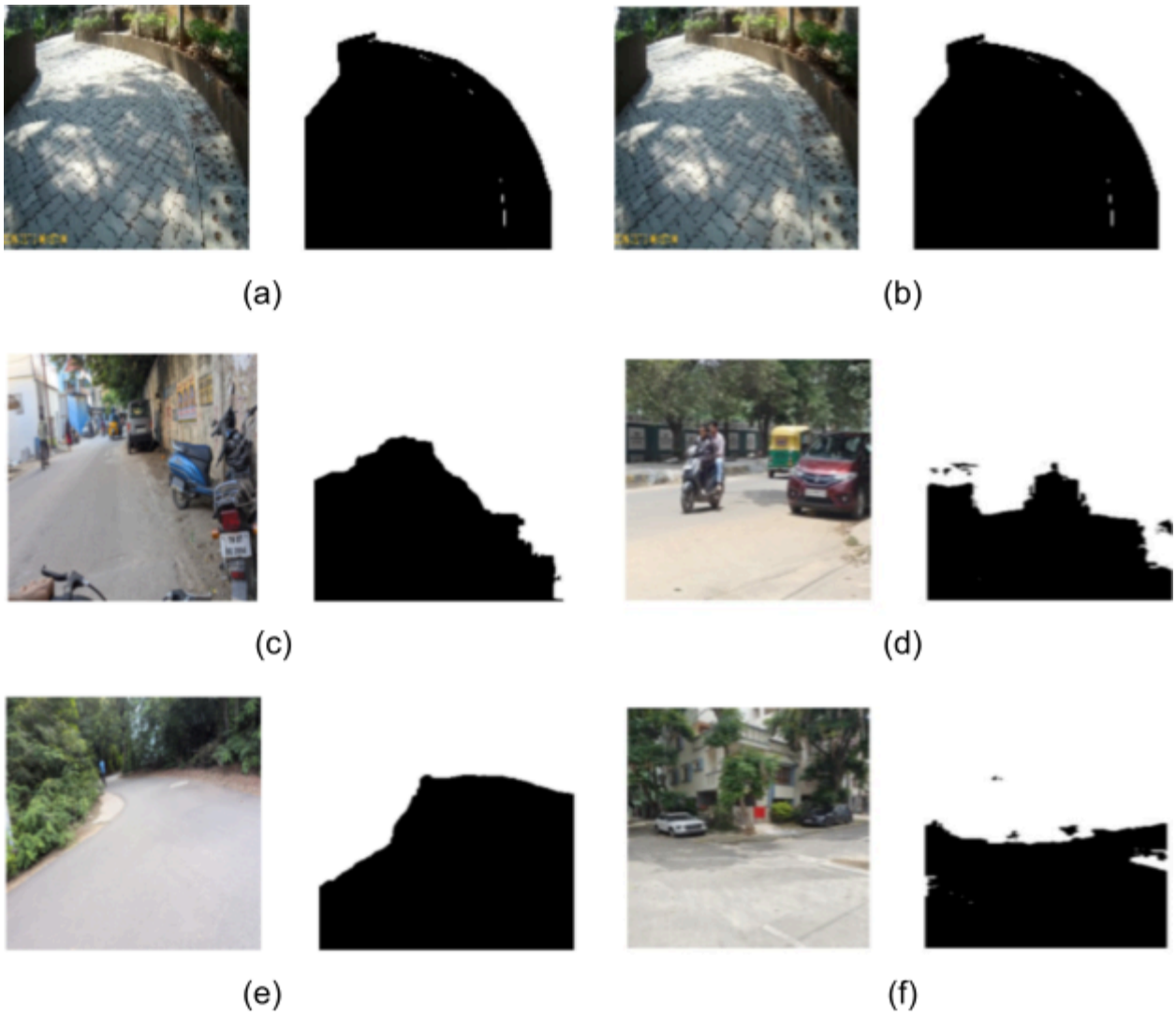


Image of roads and their segmented masks

4.3.2. Grocery Picking

4.3.2.1. Speech to Text Conversion

The first step in the grocery picking mode is to input the desired groceries from the user. This is done using a Microphone fixed in the vision module. The words are captured through the microphone and converted into text.

For the conversion of audio to text, we use **Google's speech recognition API**.

This API is preferred due to the following reasons:

- High Accuracy
- Cloud-based, no local computing needed.
- Supports multiple languages and accents.
- Easy python integration with speech recognition.
- It gave an accuracy of about **90%**

4.3.2.2. Grocery Detection

The goal of the grocery detection model is to detect all the groceries present in the image, classify them and draw bounding boxes enclosing them.

We tried with three different models - *R-CNN*, *SSD* and *YOLO*. Here is the comparison:

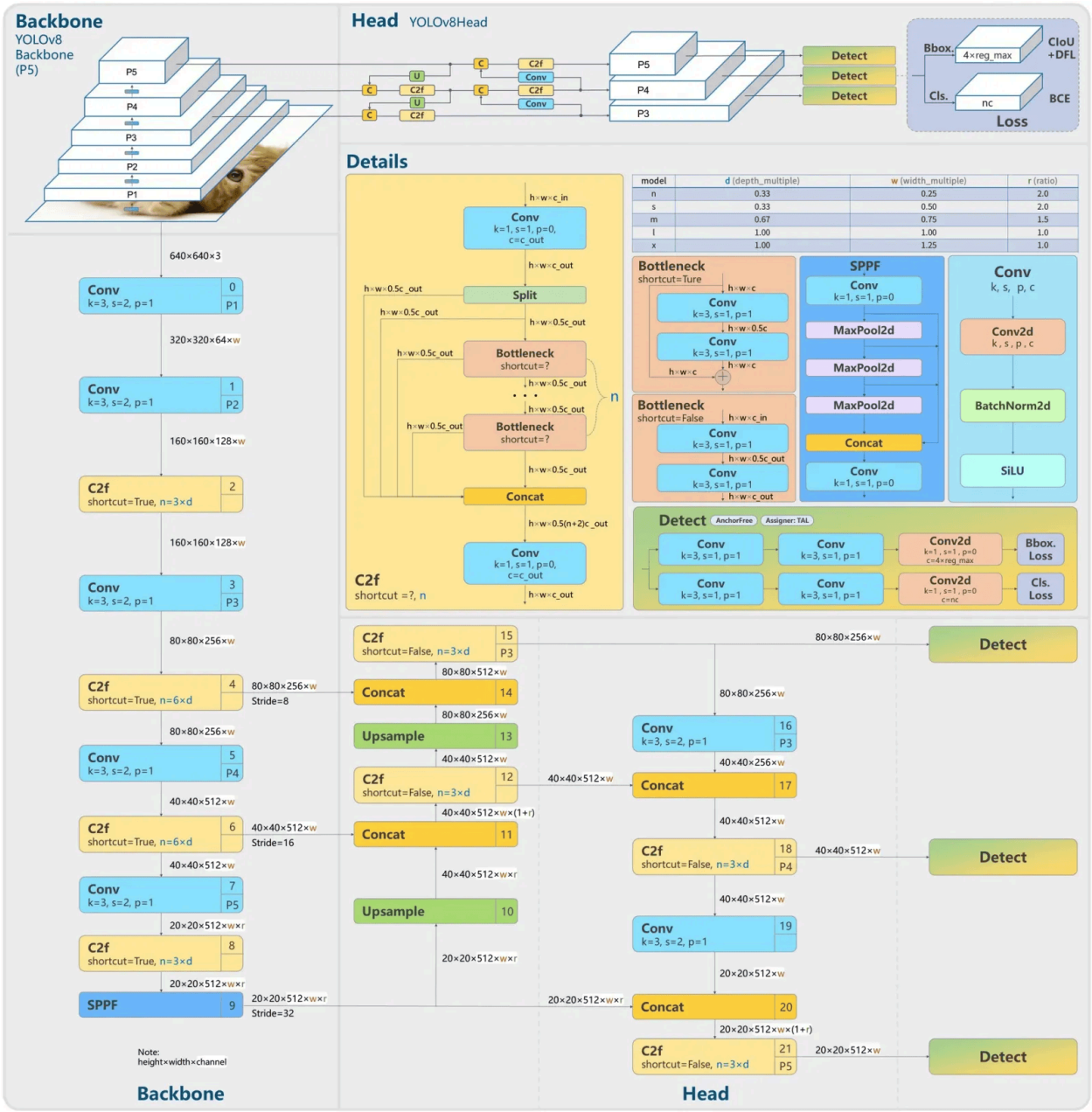
1. **R-CNN (Regions with Convolutional Neural Networks)**: R-CNN uses a relatively complex multi-stage pipeline involving - region proposal, feature extraction, classification and bounding box refinement. it processes region proposals individually through a CNN, leading to slower inference times, thus higher latency.
2. **SSD (Single shot multibox detector):** Even Though SSD is faster than R-CNN owing to the simpler semi-stage architecture, it struggles to identify and label objects correctly. Especially for small objects and overlapping objects.
3. **YOLO (You only look once)**: YOLO is purely single-stage, meaning it predicts bounding boxes and class probabilities from the input in one step. This makes it computationally lightweight, thus easy to run on edge devices. It can provide FPS much higher than

R-CNN and SSD .YOLO can detect small objects and overlapping objects effectively. Thus, iIt provides a good mix of accuracy and speed making it suitable for real-time applications.

So obviously we chose to use YOLO for grocery detection.

Among YOLO, some of the available models were YOLOv7, YOLOv8, YOLOv10 and YOLO_NAS. We used YOLOv8 by Ultralytics as it had a good balance of accuracy and speed, and is also easier to deploy on edge devices.

YOLOv8 Architecture



1. Backbone

The **backbone** is the first major part of YOLOv8 and is responsible for **extracting features** from the input image. It progressively reduces the spatial resolution while increasing the feature depth to extract hierarchical information.

Components of the Backbone:

- **Input**: The input image is resized to a fixed size **(640x640)** and normalized before being passed into the backbone.
- **Focus Layer**: Splits the input into patches to reduce spatial dimensions and capture fine details early on.
- **C2F (Coarse-to-Fine) Blocks**: These blocks split feature maps into chunks, process one part with residual bottleneck layers, and concatenate it with skipped features, resulting in multi-scale hierarchical feature learning.
- **Downsampling Layers**: The image undergoes downsampling via **stride-2 convolutions** or **pooling** layers to reduce resolution and focus on global context.
- **Output Feature Maps**: At the end of the backbone, multiple feature maps (at different resolutions) are passed to the neck for further processing.

2. Neck

The **neck** is responsible for aggregating features from the backbone and enhancing them for object detection.

Components of the Neck:

- **PANet (Path Aggregation Network)**: Combines features from different levels (low, mid, high resolutions) of the backbone to ensure multi-scale feature learning. It uses **upsampling** and **concatenation** operations to aggregate features from the higher-resolution layers with the lower-resolution layers.
- **C2F Blocks**: Used here to refine the aggregated features for better spatial and semantic representation.
- **Multi-Scale Feature Maps**: Produces three feature maps of different scales (e.g., 80x80, 40x40, 20x20 for a 640x640 input). These maps correspond to detecting small, medium, and large objects.

3. Head

The **head** is responsible for making predictions, including bounding boxes and class probabilities.

Components of the Head:

- **Anchor-Free Detection**: YOLOv8 uses an **anchor-free approach**, unlike earlier versions that relied on predefined anchor boxes. This reduces complexity and improves detection speed.
- **Prediction Layers**: The multi-scale feature maps from the neck are passed through prediction layers to generate:
    a. **Bounding Boxes**: Coordinates of detected objects.
    b. **Class Probabilities**: Confidence scores for object classes.
- **Sigmoid Activation**: Applies to the output layer to normalize predictions for class probabilities and bounding boxes.

Transfer learning and Optimization

1. **Pre-trained Model**: The YOLOv8 model, pre-trained on the extensive COCO dataset, provides a robust foundation for general-purpose object detection tasks. This makes it an ideal starting point for building the grocery-picking system..
2. **Fine Tuning on Custom Data**:The pre-trained YOLOv8 model is fine-tuned using a custom dataset of grocery items, curated by combining data from various open-source datasets and manually refining the annotations. This fine-tuning process allows the model to adjust its weights for improved recognition of objects specific to our application, enhancing both accuracy and efficiency.

   To further optimize performance, transfer learning is complemented with hyperparameter tuning. Parameters such as learning rate, momentum, and dropout rate are carefully adjusted to achieve a balance between accurate training and prevention of overfitting. This ensures the model is both efficient and well-suited for the task at hand.

Performance

The model gave an incredibly high mAP (Mean average precision) of **93%** on the test dataset indicating the ability of the model to classify objects accurately and draw precise bounding boxes.

When run in real-time with GPU acceleration, the model was able to perform grocery detection with a latency of just **11ms.**

4.3.2.3. Hand Tracking

The palm of the user has to be identified from the image to calculate the relative position of the desired grocery from the user to give appropriate instructions.

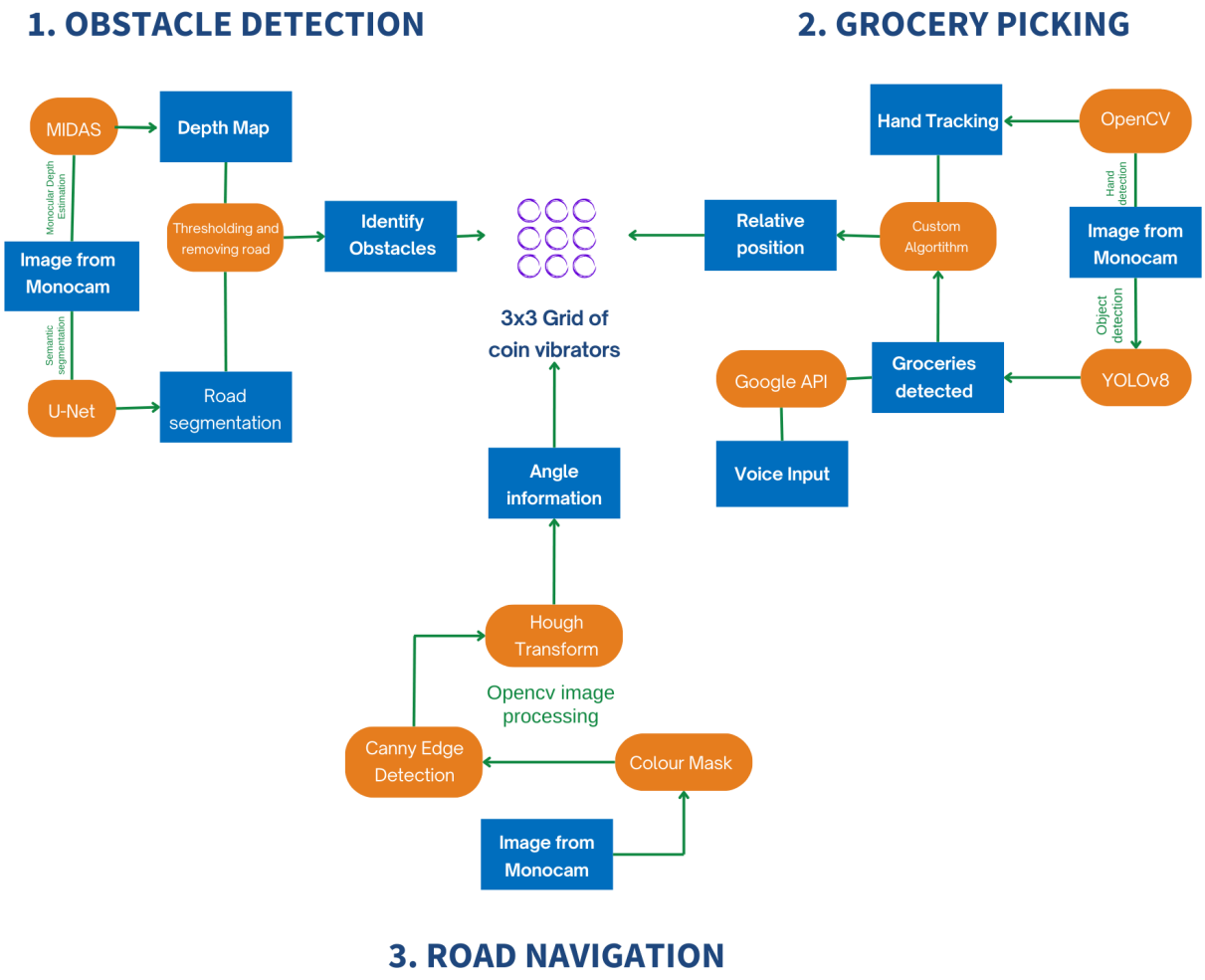
Mediapipe provides a lightweight model for hand tracking.

MediaPipe is a cross-platform framework by Google for building multimodal machine learning pipelines. Its **Hand Tracking** solution is a state-of-the-art model that detects and tracks hands in real-time with high accuracy and efficiency.

Features:

1. **21 3D Hand Landmarks:**
   MediaPipe tracks 21 key points on the hand, providing precise information about hand orientation, position, and gestures in 3D space.
2. **Real-Time Processing:**
   Designed for high-speed performance, the model can process multiple frames per second, enabling seamless real-time applications.
3. **Robustness:**
   The model is highly resilient to variations in hand sizes, orientations, and challenging environmental factors like lighting or background clutter.

The following flowchart summarizes the working of pathpal



6. Potential Applications

There are many areas where PathPal will be very useful:

- Help the visually impaired users navigate in their surroundings in their daily life, by detecting obstacles and alerting the user about them. Making them Independent in the process
- Help the user walk on roads by detecting lanes and giving a direction for the user to move in.
- Help the user pick out groceries and inform them using text to speech.
- By reducing the need for continuous human assistance, PathPal can lower the costs associated with caregiving and support services

7. Cost effectiveness analysis

The cost of the product can range from 20,000Rs to 30,000Rs. It is an affordable amount for a one time investment which will not only make the user independent, but also reduce the need for caretakers or full time assistants.

8. Challenges and Solutions

There are a few challenges that Pathpal faces:

- Latency - There might be minuscule delays that could be caused due to the microprocessor. It can be improved by looking for the latest faster processors.
- Lighting, obstacles, and unpredictable road conditions could reduce system performance and reliability. We can improve this by training the model more on real-world environments to improve the robustness of the system.
- Users might not feel comfortable shifting from using their existing white cane. We can make it easier for them by forming partnerships with assistive technology firms, **NGOs**, and **government bodies** to subsidize costs and increase accessibility.

9. Conclusion

In conclusion, the development of a visual assistance device for visually impaired individuals represents a significant step towards empowering them to live more independently and confidently. This device leverages advanced technologies to provide real-time feedback and guidance, enabling users to navigate their environment safely and efficiently. By integrating intuitive interfaces and adaptive features, the device not only addresses the practical challenges of daily life but also enhances the sense of autonomy and dignity for those affected by visual impairments.

10. References

1. https://media.geeksforgeeks.org/wp-content/uploads/20220614121231/Group14.jpg
2. https://blog.roboflow.com/content/images/size/w1600/2024/04/image-1799.webp
3. https://www.actionpro.in/wp-content/uploads/2022/08/61S7mgPXfjS._SL1112_.jpg
4. https://www.waveshare.com/media/catalog/product/cache/1/image/800x800/9df78eab33525d08d6e5fb8d27136e95/p/i/pi5-imx219-77-1_4.jpg