IDENTIFICATION OF PATHWAYS USING REALTIME VIDEO SENSOR

Aishwarya Ram Gadhave, Akhilesh Das, Renuka Rajabhau Jadhawar

I2IT Pune, Information Technology, Batch 2017, B150958517
I2IT Pune, Information Technology, Batch 2017, B150958502
I2IT Pune, Information Technology, Batch 2017, B150958523
I2IT Pune, Dept. of Information Technology

Abstract—

- **General Introduction: A microprocessor is chosen as the main controller to react towards the data received from an image sensor to give fast and accurate movement in an artificial environment. Pixel simulation and conversion of image data into mathematical arrays, performing operations on them and converting those arrays back to image programmatically is utilized.**

- **Problem Statement: To create a system that can detect pathways from given visuals and also generates insights that can be used as an input data for potential applications running on top of it.**

Index Terms—Robotics, Computer Graphics, Data Analysis.

I. INTRODUCTION

A. General Overview:

Our supposedly (relatively) low cost mobile robot construction is intended to serve as a good example and motivation source, on which anyone can jump start their computer vision projects. Our ideology is the way of tackling our own self-defined problem, our approach, and our vision to create the end product.

B. Idea Overview:

We often encounter many discoveries and applications which rely on either artificial intelligence or machine learning these days. While they serve many great purposes, we feel that there's another aspect of growing technology which we simply can't refuse. And that is Computer Vision.

Vision-based projects also have so much potential yet to be discovered, especially at times where human reaction time is the most limiting factor when evaluated. We are attempting to be the torchbearers of this aspect, and want to encourage fellow developers to work on this domain.

Here, we will be implementing a very simple, easy to use, easy to understand path recognition system that'll check its surroundings and guide a robot on it accordingly. This project is built in such a manner that others can readily take up the code base and start implementing their own features too.

II. LITERATURE SURVEY

1) Title of the Paper:

A simple and robust line detection algorithm based on small eigenvalue analysis.

Authors: D.S Guru, B.H Shekar, P Nagabhushan.

Year: 2004

Observations and Gatherings: The publication focused on detecting line from pixels through advanced mathematical formulae. We found out that how they achieved their goal, can be simplified to provide simplicity in our application of the same thing.

2) Title of the Paper:

Parallel Hough Transform-Based Straight Line Detection and Its FPGA Implementation in Embedded Vision.

Authors: Xiaofeng Lu, Li Song, Sumin Shen, Kang He.

Year: 2013

Observations and Gatherings: The authors used Hough Transform of pixel intensities to capture possible line segments. Though the paper focuses too much on electronics (our goal is digital), it still proved to be a great starting point to understand line detection for us.

3) Title of the Paper:

Design and Implementation of Line Follower Robot.

Authors: IEEE.

Year: 2010

Observations and Gatherings: Another great source of climbing basic ropes of computer vision based projects.

4) Title of the Paper:

A low cost mobile robot for engineering education.

Authors: IEEE.

Year: 2006

Observations and Gatherings: This publication served as a great head-start for considering financial targets of our project. It still lacked details about expanding the complexity of the robot in our opinion.

III. PROPOSED SYSTEM

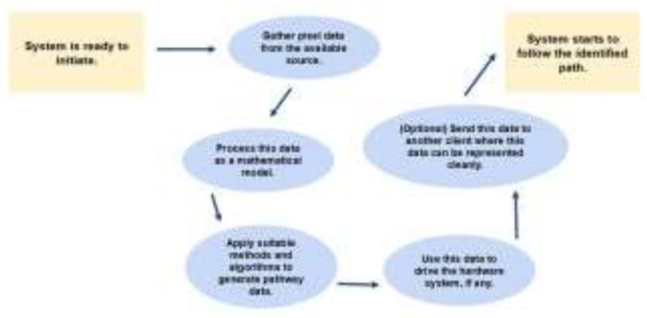

System is ready to initiate.
Gather pixel data from the available source.
Process this data as a mathematical model.
Apply suitable methods and algorithms to generate pathway data.
Use this data to drive the hardware system, if any.
(Optional) Send this data to another client where this data can be represented cleanly.
System starts to follow the identified path.

Fig. 1: Proposed system.

A. System Overview:

• Able to track white surfaces without any setup. The robot is able to detect neighbouring or any other colour values, but it many require manual tuning before actually traversing a random path.

• Able to detect indicate path, its steepness and its curliness through a simple camera input.

• Orchestrates the whole hardware setup attached to its CPU to walk on the deducted pathway.

B. Module Explanation:

• GUI Controls: (Optional) a module dedicated to representing ongoing processes, their outcomes, and other controls that makes understanding the ongoing calculations on the robot very easy and intuitive.

• Hardware Assembly: A module which contains the brain of the whole assembly. Includes the microprocessor, along with the necessary electronic components that makes it ”move” and ”see”.

• Software: Includes everything that runs in the microprocessor. Base language interpreter, libraries, macros and methods, and our business logic.

C. Techniques and Algorithms:

• Pixel Summation.

• Conversion of image data into mathematical data and vice versa.

• Mathematical operations which result into image transformations.

IV. NON FUNCTIONAL REQUIREMENTS

- **Performance Requirements:** For good performance, the runtime environment should be tuned to mission critical processes and most of the RAM should be used for our application.

- **Safety Requirements:** Proper monitoring of the hardware components, such as overheating, short circuits must be avoided and should be kept an keen eye on to avoid damaging the implementation or its desired outcomes.

- **Software Quality Attributes:** Application implementation will satisfy following software quality attributes:

– Version controllable through Git.

– Neatly presented code base with necessary comments baked into the code.

– Easily extensible and testable.

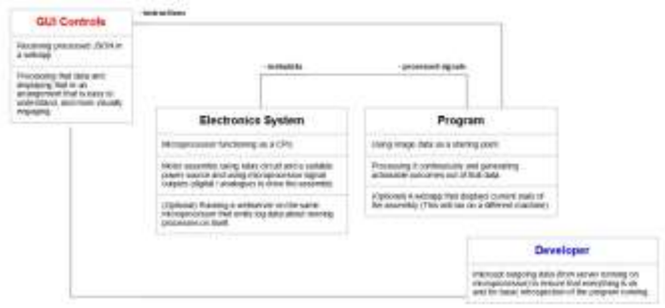

GUI Controls
Electronics System
Microprocessor functioning as a CPU
Program
Using image data as a starting point
Processing it contextually and generating actionable outcomes out of that data
Developer

Fig. 2: Proposed module communication protocol.

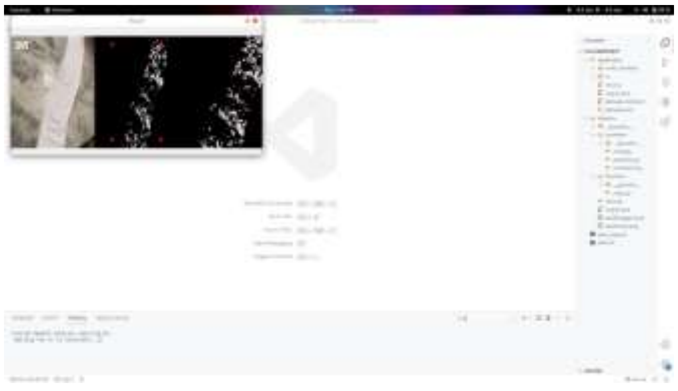
V. RESULTS

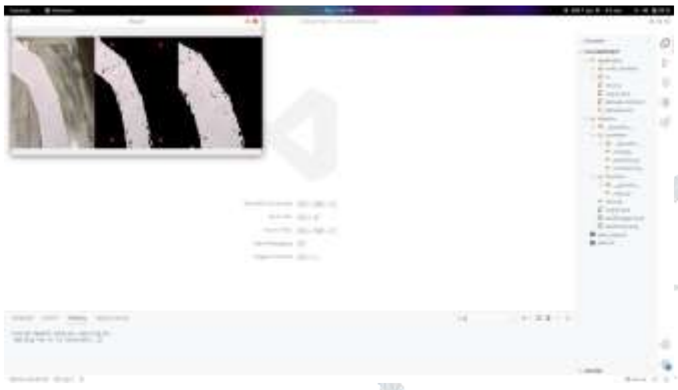
- Running a custom script that facilitates easy execution of project features, from running something to moving files in the required directory.

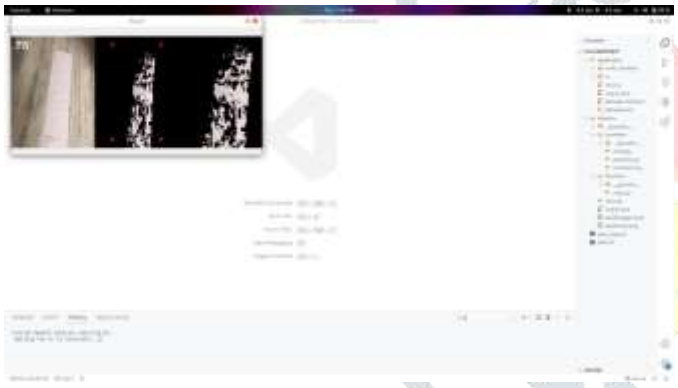
- Main OpenCV program analysing image input, processing in collecting valuable insights and saving into external text file.

- Moving the output text file and running demo application, all handled by the automation script.

- Demo app running on the data generated previously by the OpenCV program.







VI. CONCLUSION

Conclusion:

We were able to research about existing path following techniques and hardware available, and enhance it with the ability of real-time computer vision to tackle even more complex paths it is exposed to, and come up with our own implementation that is simple, easy-to-work, modifiable and performant under simulated environment.

Assumptions Taken While Deriving This Conclusion:

- The final assembly (robot) will be subjected to an artificial environment, and is not guaranteed to run outside that environment as expected.
- Fine tuning of robot may be required after multiple test / production runs.

REFERENCES

[1] "Python (programming language)", June 4, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Python (programming language) .

[2] “pip (package manager)”, May 2, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Pip (package manager) .

[3] “Visual Studio Code”, June 3, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Visual Studio Code.

[4] D.S Guru, B.H Shekar, P Nagabhushan. A simple and robust line detection algorithm based on small eigenvalue analysis. 2004.

[5] X. Xiaofeng Lu, Li Song, Sumin Shen, Kang He. Parallel Hough Transform Based Straight Line Detection and Its FPGA Implementation in Embedded Vision. 2013.

[6] IEEE. Design and Implementation of Line Follower Robot. 2010.

[7] IEEE. A low cost mobile robot for engineering education. 2006