Artificial Intelligence based Autonomous Robot

Subramani Narayanaswamy, Sudarshan Sridharan, Varun Shijo, Neena Jacob

Abstract— The aim of this project is to create a proof-of-concept model for an autonomy framework for robots using artificial intelligence. Reinforcement learning methods will be implemented by making use of models incorporating artificial neural networks. Conventionally, obstacle avoidance has been implemented using a procedural approach that makes use of predefined hard coded values; that are arrived at by means of trial and error. To eliminate the need for inaccurate and hard-to-determine fixed threshold values, we use Artificial Intelligence. Initially, there will be zero knowledge present, and unsupervised learning will be performed. The robot will be set to default to forward movement until it collides with an obstacle while maintaining a buffer of all sensor readings at all times. It then reads from the buffer every time a collision is made to learn the conditions at which collisions occur. The safe distance which was initialized to zero keeps getting incremented by the learning rate 'a'. Optimal path to avoid obstacle collision can be extrapolated by calculating the local maximum of the longest bitonic sub sequence in the array of distance readings taken in a sweep of the servo with the ultrasonic sensor mounted on top.

Index Terms— Artificial Intelligence, Autonomous Robot, Collision avoidance, Hardware agnostic, Longest Bitonic Sub sequence, Perceptron, Weights

1 Introduction

The project will employ concepts from Digital Signal Processing, Dynamic Programming for path calculation and Artificial Neural Networks for Machine Learning as a part of Artificial Intelligence.

Currently, most hardware oriented applications consist of a procedural approach towards implementation of autonomy. However, the procedural approach can only be used to implement Artificial Narrow Intelligence. We aim to initiate a transition towards Artificial General Intelligence by expanding upon the avenues to which ANI is being applied and coalescing these into a step in the direction of AGI.

2 Need Of Project

A robot is a container for AI, sometimes mimicking the human form, and sometimes not; but the AI itself is in the computer inside the robot. AI determines the governing behavior, and the robot is merely its body. For example, Apple's Siri is AI, the woman's voice we hear is a personification of that AI, and there's no robot involved at all.

There are many different types or forms of AI since AI is a broad concept, the critical categories we need to think about are based on an AI's caliber. There are three major AI caliber categories:

- **Artificial Narrow Intelligence (ANI)**

Sometimes referred to as Weak AI, Artificial Narrow Intelligence is AI that specializes in one area. There's AI that can beat the world chess champion in chess, but that's the only thing it does. Ask it to figure out a better way to store data on a hard drive, and it'll look at you blankly[4].

- **Artificial General Intelligence (AGI)**

Sometimes referred to as Strong AI, or Human-Level AI, Artificial General Intelligence refers to a computer that is as smart as a human across the board a machine that can perform any intellectual task that a human being can. Creating AGI is a much harder task than creating ANI, and we're yet to do it. Professor Linda Gottfredson describes intelligence as a very general mental capability that, among other things, involves the ability to reason, plan, solve problems, think abstractly, comprehend complex ideas, learn quickly, and learn from experience. AGI would be able to do all of those things as easily as you can[4].

- **Artificial Superintelligence (ASI)**

Oxford philosopher and leading AI thinker Nick Bostrom defines superintelligence as an intellect that is much smarter than the best human brains in practically every field, including scientific creativity, general wisdom and social skills. Artificial Superintelligence ranges from a computer that's just a little smarter than a human to one that's trillions of times smarter across the board. ASI is the reason the topic of AI is such a spicy meatball and why the words immortality and extinction will both appear in these posts multiple times [4].

2.1 Existing System

The existing implementations of autonomous robots consist of the developer team, using trial and error, determining threshold values and manually tuning the behavior of the system as required. This is a very tedious process and is specific to the form factor of the container. When the container parameters change, the values must be tuned again. For instance, in the form factor of a car, if the axial length is changed, the turning radius changes. So a behavioral model created for one car will not work for a slightly different car, out-of-the-box.

2.2 Proposed System

The proposed system will contribute towards the idea of Artificial General Intelligence, as the project aims at using reinforcement learning methods like regression, perceptron, etc for collision detection. Human intervention or hard-coded programming and values will not be necessary to guide the robot to adapt to the environment. The implementation of the system would not be restricted to just this form factor of a robot, but it aims to be implemented in any hardware framework and environment, for exam-

ple, a monster truck. Though the placement of the sensors and IMU would be different in the robot and the monster truck, as per the size of the form factor, the system will adapt to the framework based on the readings and data collected from the robot in a generalized form[1],[2].

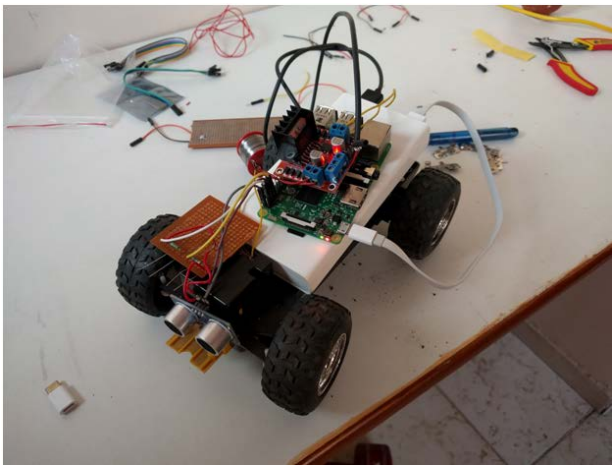

Fig. 1. Hardware Container (Robot car)

3 Literature Survey

- **Unsupervised Learning**

Unsupervised machine learning is the machine learning task of inferring a function to describe hidden structure from "unlabeled" data (a classification or categorization is not included in the observations). Since the examples given to the learner are unlabeled, there is no evaluation of the accuracy of the structure that is output by the relevant algorithm—which is one way of distinguishing unsupervised learning from supervised learning and reinforcement learning.
A central case of unsupervised learning is the problem of density estimation in statistics, though unsupervised learning encompasses many other problems (and solutions) involving summarizing and explaining key features of the data[4].

- **Artificial Neural Networks**

Artificial Neural Networks are a computational approach which is based on a large collection of neural units loosely modeling the way the brain solves problems with large clusters of biological neurons connected by axons. Each neural unit is connected with many others, and links can be enforcing or inhibitory in their effect on the activation state of connected neural units. Each individual neural unit may have a summation function which combines the values of all its inputs together. There may be a threshold function or limiting function on each connection and on the unit itself such that it must surpass it before it can propagate to other neurons. These systems are self-learning and trained rather than explicitly programmed and excel in areas where the solution is difficult in a traditional computer program [5],[6],[7].

- **Perceptron**

The perceptron algorithm; its first implementation, in custom hardware, was one of the first artificial neural networks to be produced. In machine learning, the perceptron is an algorithm for supervised learning of binary classifiers. It is a type of linear classifier, i.e. a classification algorithm that makes its predictions based on a linear predictor function combining a set of weights with the feature vector. The activation function of perceptron is a function like The Heaviside Step function (Figure 2) which returns '1' if the input is positive or zero, and '0' for any negative input [8]. Detailed study on Perceptron and its implementation can be found in [8].

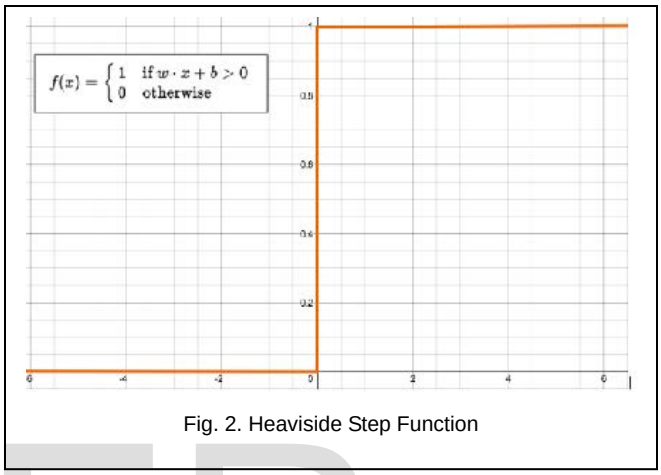

Fig. 2. Heaviside Step Function

- **Longest Bitonic Sequence**

A bitonic subsequence is a subsequence that is first increasing up to a peak value and then decreasing from the peak value. For example, A=[1, 11, 2, 10, 4, 5, 2, 1] the longest bitonic sequence is 1, 2, 4, 5, 2, 1 of length 6. For A=[0, 8, 4, 12, 2, 10, 6, 14, 1, 9, 5, 13, 3, 11, 7, 15] longest bitonic sequence is 0, 8, 12, 14, 13, 11, 7 of length 7.
Note that a bitonic sequence starting from a value reaches a peak value in a strict increasing order of values. Then it starts decreasing monotonically. So, we can easily perceive that a bitonic sequence consists of a increasing subsequence and a decreasing subsequence. So, a longest bitonic subsequence would be subsequence that consists of a longest increasing subsequence (LIS) ending at peak and a longest decreasing subsequence (LDS) starting at peak[3].

4 Methodology

- **Crash based Learning Module**

The objective of the agent is to avoid crashing without using hard-coded predefined values for stopping distance from the nearest obstacle. This is achieved by learning from initial crashes and tuning the synaptic weights to change in subsequent iterations the stopping distance.
Initially unsupervised learning is performed by the system where the robot on crashing into an obstacle, backtracks to the previous safe state, increments the current Threshold value (initially set to zero) by the learning rate and dumps the new state of the system. Optimal path calculation is then done by taking a sweep of the environment using Ultrasonic sensor mounted on a servo motor. The readings are plotted in the

form of a bell shaped curve, to find the longest bitonic subsequence from the array of readings. This longest bitonic array will shows the most promising direction or path for the robot's movement. This is shown in Fig. 3.

- **Optimal Path Calculation and Collision Avoidance Module**

Prediction of crash is done by making use of a non-linear perceptron, which factors in distance to the wall, to predict the braking distance where the motor power is cut-off. For each instant of distance weights to the obstacle are compared to the tuned threshold value. If the distance weight at an instant is greater than the tuned threshold, then the robot continues to move in a forward direction, else the robot stops and performs Optical Path Calculation to reroute for a collision free movement. This is shown in Fig. 4.

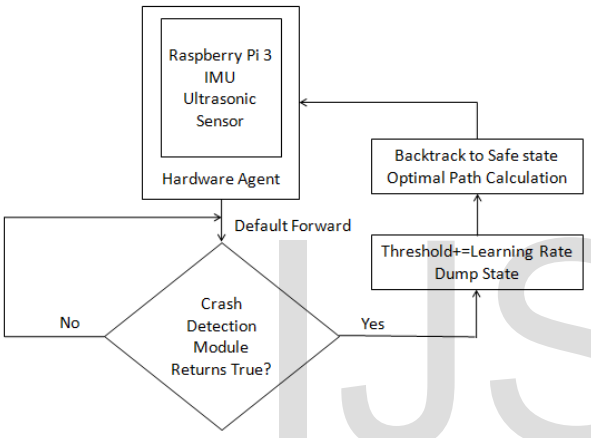

Fig. 3. Crash based Learning Module

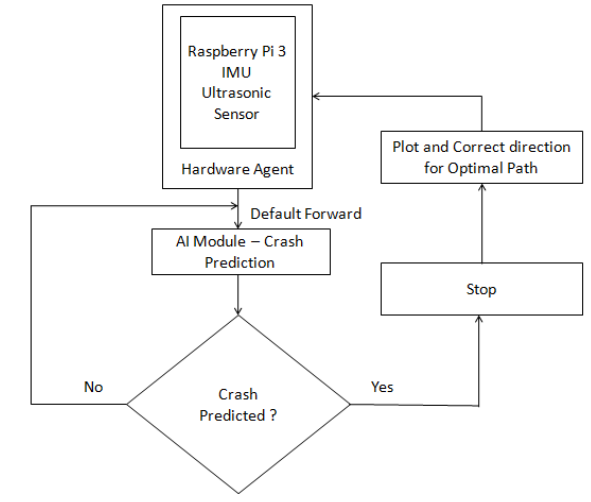

Fig. 4. Optimal Path Calculation and Collision Avoidance Module

5 Experimental Results

The hardware agent or robot is initially given a default forward. When it collides with an obstacle on its path the threshold values are tuned. The agent backtracks to the safe state once the collision has detected. Figure 5 shows the distance sensing at intervals with initial threshold as zero and a random weight of 0.5. The learning rate 'a' or here 'alpha' is set to 15. Lower the learning rate, the more accurate prediction of collision, but with more number of iterations.
Optimal path calculation module performs the longest bitonic processing of the distance measures obtained from the ultrasonic sensor to detect the most promising path for the robot to move.

```
{'threshold': 0, 'alpha': 15, 'weight': 0.5}
{'threshold': 0, 'alpha': 15, 'weight': 0.5}
{'threshold': 0, 'alpha': 15, 'weight': 0.5}
{'threshold': 0, 'alpha': 15, 'weight': 0.5}
Crash detected. Tuning Weights and rerouting.
```

Fig. 5. Initial Threshold

The distance array readings for one sweep of the ultrasonic sensor are as follows:

```
Crash Detected. Tuning weights and rerouting
Distance Array:
87.39
70.52
100.0
100.0
94.76
99.26
88.39
99.09
90.3
96.69
100.0
83.36
99.04
100.0
96.91
100.0
Optimal Path: Right
```

Fig. 6. Distances Sweel and Optimal Path Calculation

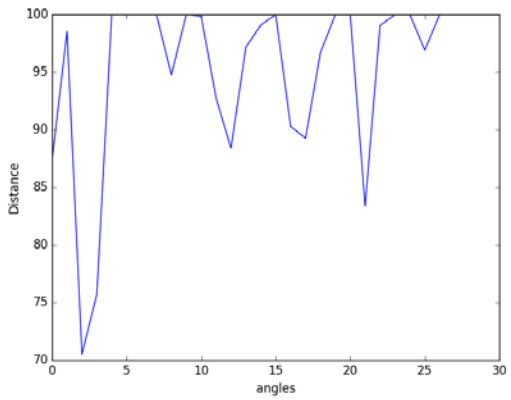

Fig. 7. Graph of Ultrasonic Sweep readings

On crashing, the threshold is increased by the learning rate for the next iteration of distance sensing and collision prediction. This is shown in Figure 8.

```
{'threshold': 15, 'alpha': 15, 'weight': 0.5}
{'threshold': 15, 'alpha': 15, 'weight': 0.5}
{'threshold': 15, 'alpha': 15, 'weight': 0.5}
{'threshold': 15, 'alpha': 15, 'weight': 0.5}
Crash detected. Tuning Weights and rerouting.
```

Fig. 8. Threshold Tuning

The threshold is tuned in this way each time a collision is detected. Once the threshold is tuned to the appropriate value, then crash is predicted and an optimal path where the robot can move default forward without collision is decided. One such example after 3 collisions is shown in Figure 9.

```
{'threshold': 45, 'alpha': 15, 'weight': 0.5}
{'threshold': 45, 'alpha': 15, 'weight': 0.5}
{'threshold': 45, 'alpha': 15, 'weight': 0.5}
{'threshold': 45, 'alpha': 15, 'weight': 0.5}
{'threshold': 45, 'alpha': 15, 'weight': 0.5}
Crash Predicted. Commencing pre-emptive rerouting.
Optimal Path: Right
```

Fig. 9. Crash Prediction and Optimal Path Calculation

6 Conclusion and Future Scope

The project gives an autonomous robot capable of making real-time decisions without human intervention. The project's fruition results in a modular framework that can be expanded to incorporate a multitude of form factors by abstracting the form factor's influence on the agent from the method of learning and adapting to itself and its environment. This is achieved by keeping a standard minimum sensor/actuator skeleton which can be scaled according to need.

The objective of creating a system logic that can be implemented to any form factor in the form of Artificial Intelligence, is displayed by collision avoidance by using weights that determine whether the system is in a positive or negative state. This is achieved by using the various reinforcement learning methods to train the robot and make it understand the environment. Thus, this system gives a platform for any hardware framework to be implemented to avoid collision.

Future Scope:

- The project can be used for interfacing the system in many other technologies and hardware frameworks.
- Speed breakers can be crossed with adaptive speed and variable PWM can be given for climbing slopes at a constant speed.
- The project will contribute to social aspects by being implemented in applications like Automated Wheelchair, Accident free roads, etc.
- Self preservation in the vein of Isaac Asimov's laws of robotics can be implemented, for example, by letting the robot disobey a user command that puts the robot in the way of harm.
- To incorporate image and speech processing modules for input.

Acknowledgment

The authors would like to take this opportunity to thank all those individuals who have helped us in visualizing this project. We express our deep gratitude to our project guide Prof. Neena Jacob, who is also the Project Coordinator, for providing timely assistance to our queries guidance that she gave owing to her interest in the field of Artificial Intelligence. We extend our sincere appreciation to all our professors from the SIES Graduate School of Technology for their valuable inside during the designing and presentation of the project. Their contributions have been helpful in building the idea of our project. We are also grateful to our HOD-IT Dept, Prof. Leena Ladge for extending her support towards the project.

References

[1] K. Sreevishakh and S. Dhanure, *A Review Paper on Automotive Crash Prediction and Notification Technologies*, International Conference on Computing Communication Control and Automation (ICCUBEA), 2015.

[2] Gehrig and F. Stein, *Collision Avoidance for Vehicle-Following Systems*, IEEE transactions on intelligent transportation systems, Volume 8, 2007.

[3] K. Chiew and S. Qin, *Scheduling and Routing of AMOs in an Intelligent Transport System*, IEEE Transactions on Intelligent Transportation Systems, Volume 10, 2009.

[4] Tim Urban. *The AI Revolution: The Road to Superintelligence* @ONLINE. Jan 2015. URL: http://waitbutwhy.com/2015/01/artificial-intelligence-revolution-1.html.

[5] Andrej Karpathy. *Hacker's guide to Neural Networks* @ONLINE. URL: http://karpathy.github.io/neuralnets/.

[6] Andrew Ng. *Machine Learning by Stanford University* @ONLINE. URL: https://www.coursera.org/learn/machine-learning/.

[7] Daniel Shiffman. *The Nature of Code, Chapter 10. Neural Networks* @ONLINE. URL: http://natureofcode.com/book/chapter-10-neural-networks/.

[8] *Perceptrons - the most basic form of a neural network* @ONLINE. June 2016. URL: https://appliedgo.net/perceptron/.

[9] Wang Zheng. OpenCV Python Neural Network Autonomous RC Car [Video file]. (2011, June 11). URL: https://www.youtube.com/watch?v=BBwEF6WBUQs.

IJSER