Detection and Avoidance of Simulated Potholes in Autonomous Vehicle Navigation in an Unstructured Environment

Jaiganesh Karuppuswamy, Vishnuvardhanaraj Selvaraj, Meyyappa Murugappa Ganesh and Ernest L Hall

Center for Robotics Research
University of Cincinnati
Cincinnati, OH 45221-0072

ABSTRACT

In the navigation of an autonomous vehicle, tracking and avoidance of the obstacles presents an interesting problem, as this involves the integration of the vision and the motion systems. In an unstructured environment, the problem becomes much more severe as the obstacles have to be clearly recognized for any decisive action to be taken.

In this paper, we discuss a solution to detection and avoidance of simulated potholes in the path of an autonomous vehicle operating in an unstructured environment. Pothole avoidance may be considered similar to other obstacle avoidance except that the potholes are depressions rather than extrusions from a surface. A non-contact vision approach has been taken since potholes usually are significantly different visually from a background surface. Large potholes more than 2 feet in diameter will be detected. Furthermore, only white potholes will be detected on a background of grass, asphalt, sand or green painted bridges.

The signals from the environment are captured by the vehicle’s vision systems and pre-processed appropriately. A histogram is used to determine a brightness threshold to determine if a pothole is within the field of view. Then, a binary image is formed. Regions are then detected in the binary image. Regions that have a diameter close to 2 feet and a ratio of circumference to diameter close to pi are considered potholes. The neuro-fuzzy logic controller where navigational strategies are evaluated uses these signals to decide a final course of navigation.

The primary significance of the solution is that it is interfaced seamlessly into the existing central logic controller. The solution can also be easily extended to detect and avoid any two dimensional shape.

Keywords: computer vision, pothole detection, object location, mobile robot, blob analysis.

Correnspondence: Jaiganesh Karuppuswamy, Email: kjaiiagesh@hotmail.com, WWW: http://www.robotics.uc.edu, Phone: (513) 556-2730, FAX: (513) 556-3390.
1. INTRODUCTION

Humans have sophisticated vision systems through which a large amount of information can be received and interpreted very quickly. We also have extremely adept systems for making decisions based on these visual input and also the means to perform appropriate actions. A truly autonomous robot must sense its environment accurately and react appropriately. This issue attains greater importance in an outdoor, variable environment.

The Center for Robotics Research at the University of Cincinnati has been involved in building such an autonomous robot for competing at the International Ground Robotics Contest conducted by the Association for Unmanned Vehicles System International. In the main category of the contest, the robot has to navigate around an outdoor obstacle course. Two white/yellow lines spaced 10 feet apart that may be either continuous or discontinuous mark the course. The obstacles placed in the course are 10 gallon white buckets and construction drums.

At the competition for this year, a new obstacle was introduced in the navigation course – simulated potholes. These are white circles of size 2 feet diameter placed randomly across the course. The purpose of this paper is to describe our paradigm for detecting these potholes.

The existing vision and motion control systems are discussed in Section 2. Section 3 describes the schema for the new vision system. The design of physical parameters for the system is covered in Section 4. Section 5 describes the necessity and the selection of an appropriate imaging board for the solution. The methodology adopted for the detection of simulated potholes and the detailed steps are described in Section 6. Section 7 describes the design of the software. Finally, Section 8 details the integration of the solution with the existing Bearcat II robot.

Design Objectives

![Figure 1 A simulation of the potholes in the course.](image)

The primary objective of this research is to arm the robot with the functionalities required to successfully detect and avoid any simulated potholes in its course. The design of the solution
for this problem was kept in parallel with the design philosophy of the Bearcat II robot – to keep it an off-the-shelf solution and to attack the problem in a ‘divide-and-conquer’ approach.

A simulation of the pothole in the course is shown in Fig. 1. These are only some of the possible scenarios that can occur and they largely depend on the orientation of the robot with respect to the course.

2. EXISTING SYSTEM CONFIGURATION

The development of the Bearcat II is based upon the realization that the design of a complex electro-mechanical system, like an automated guided vehicle, must be accomplished by a decomposition of the design problem into simpler units. This decomposition would be carried on until all units reach the individual component level. These components can then be designed, integrated to form major sub-units and ultimately, on further integration, lead to the entire system. The major subsystems in the Bearcat II robot are vision system, obstacle avoidance system, steering control system, speed control system, safety system and central logic controller.

Vision Solution

![Diagram of Vision Solution](image)

Figure 2 Existing vision solution.

Two JVC CCD cameras are used to view and follow the left or the right lane marker. Only one lane marker is followed at any instant. A CCSU-8BW video switch device from FSR, Inc. alternates between the two cameras depending on the visibility of the lane marker. The central logic controller (CLC) controls the video switch through the Galil DMC.

An ISCAN RK-446-R image-tracking device processes the image from the camera. The device finds the centroid of the brightest or darkest region in a computer controlled window, and returns the X, Y co-ordinates of its centroid and the size information of the blob. If no object is found, a loss of track (LOT) signal is generated. The cameras are angled downward at 32° and
panned to the front at 30°. This setup gives a 4-foot wide view of the ground and 6-foot view ahead.

Image co-ordinates are two-dimensional while actual world co-ordinates are three-dimensional. In an autonomous situation, the problem is to determine the three-dimensional coordinates of a point on the line given its image coordinates. A new algorithm [1] was developed to establish the mathematical and geometrical relationships between the physical 3-D world coordinates of the line to be followed and its corresponding 2-D digitized image coordinates. The mean square error between these measured and computed points was 0.242 inches for the X-axis and 0.295 inches for the Y-axis, which established the accuracy of the algorithm. A new simpler vision calibration method has also been designed which uses only four 3-D points.

Motion Control Solution

Two individually driven front wheels powered by two 36-volt, 15-amp motors are used. The motors drive the left and the right wheel separately through two independent gearboxes, which increase the motor torque by a factor of 40. This enables a zero turning radius by rotating one wheel in the forward direction and the other in the reverse direction. This unique design offers the ability to make a turn about the center of axis of the drive wheels thereby providing the vehicle exceptional maneuverability. The power to each motor is delivered from an AMC DC 48A amplifier that amplifies the signal from the Galil DMC Motion Controller. To complete the control loops; a position encoder is mounted on each of the drive motors. A castor wheel at the back of the vehicle balances the load.

Steering the vehicle is achieved by varying the speed of the left and the right wheels while negotiating a curve. This enables the vehicle to make a curved turning path parallel to the track lines. By manipulating the sum and difference of the speed of left and the right wheels, the velocity and the orientation of the vehicle can be controlled at any instant.

\[
\begin{align*}
\text{Velocity of Vehicle,} & \quad V_{\text{avg}} = \left( V_L + V_R \right) / 2 \\
\text{Orientation of Vehicle,} & \quad \theta = \left( V_L - V_R \right) / WT
\end{align*}
\]

where,  
VL = Velocity of the left wheel  
VR = Velocity of the right wheel  
W = Distance between the center of the two wheels  
T = Sampling time

The design objective in selecting the motor control parameters was to obtain a stable control over the steering system with a good phase and gain margin and a fast unit step response. For this purpose a Galil Motion Control Board with a Proportional Integral Derivative controller (PID Controller) was used. The system was modeled in MATLAB using SIMULINK and the three parameters of the PID controller were selected using a simulation model to obtain optimum response. The unit step response values for the PID controller were tested on the actual vehicle and were fine tuned using the software supplied by Galil Motion, Inc. – WSDK 1000.
3. DESIGN SOLUTION

The schematic of the solution for detection of simulated potholes is illustrated in Fig.3. As painting white circles on the course simulates the potholes, vision system is the only solution to detect the potholes. At the same time, the existing two cameras cannot be used for this purpose as they were tied up in the line following system. A monochrome Panasonic CCD camera is used to capture the course ahead of the robot. A monochrome camera was selected because then the data would be compact compared to a color camera. The data from the camera is fed into the video switch and the central logic controller is used to select a single camera for data processing.

![Schematic of the solution for detection of simulated potholes]

Figure 3. Design solution uses an additional camera.

4. DESIGN OF PHYSICAL PARAMETERS

The location and orientation of the camera plays a vital role in the correct functioning of the pothole detection system. The orientation parameters determine the view of the camera and consequently the size and shape of the resulting image of the simulated pothole. One of the important considerations during the design of the physical parameters for the camera is the reaction distance $[2]$ of the robot. This distance is the minimum distance within which an obstacle needs to be detected for the robot to successfully avoid it.

The reaction distance for the Bearcat II robot is 12 feet. This means that the pothole must be detected at a distance no less than 12 feet for the robot to successfully maneuver away from it. The camera should be located at the highest point to have the largest possible field-of-view. The present design of the robot allows the camera to be placed at a height of 4 feet. Based on these physical parameters, the angle at which the camera needs to be oriented is calculated to be 71°.

The camera also needs to be calibrated in order to establish a frame of reference between the real world coordinates and the image coordinates. The calibration process for the camera
consists of determining the permissible error range for the size of the simulated pothole in the image.

5. IMAGING BOARD

The data from the Panasonic camera is directly passed to the central computer through an imaging board. The reasons for using an Imaging Board as an intermediary between the central logic controller and the camera are:

1. The imaging board reduces the computational overheads required for complex imaging calculations at the central logic controller.
2. The image manipulation routines used by the imaging board are compact and efficient.
3. Off-the-shelf library functions are available for image manipulation, which can reduce the solution development time significantly.

Selection of Imaging Board

A number of constraints needed to be born in mind before a suitable imaging board could be selected for the application. These are:

Operating System: The robot is built over the DOS operating system and most of the hardware in the robot use the DOS interrupt vectors for communication with the Central Logic Controller. Hence the imaging board needs to be compatible with the DOS operating system.

PCI Host Interface: The existing hardware on the robot use all the available ISA and USB ports in the computer motherboard. So the imaging board needed to be interfaced through a PCI Host Interface.

Monochrome Input: As the data comes from a monochrome CCD camera, the imaging board had to be monochrome-compatible.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Meteor</th>
<th>DT3150</th>
<th>PX610</th>
<th>SV4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Board Cost</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Camera Input</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>DOS Compatibility</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Library Support</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Spatial Resolution</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Onboard Memory</td>
<td>8</td>
<td>4</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>Data Transfer Rate</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Total Score</td>
<td>61</td>
<td>58</td>
<td>51</td>
<td>80</td>
</tr>
</tbody>
</table>

Based on these initial constraints, four imaging boards were selected – Meteor of Matrox, Inc., DT3150 of Data Translation, PX610 of Imagenation and PIXCI SV4 of EPIX, Inc. A
A decision matrix was then developed to select one among these four Imaging Boards based on chosen selection criteria. The decision matrix that was developed is illustrated in Figure 4.

Based on the decision matrix, the PIXCI SV4 Imaging Board manufactured by EPIX, Inc., was selected for the solution.

6. SOLUTION METHODOLOGY

The methodology for detecting the presence of a pothole consists of a series of imaging operations[^3] performed on the image obtained from the camera. All these operations are performed at the imaging board to reduce the computational load on the central computer that controls the robot.

As it is with any image processing solution, the raw image from the camera needs to be pre-process before any operations can be performed over it. The data from the camera is in the form of an array of pixel values, with each of the pixel represented in a gray scale value ranging between 0 and 255. This image is captured by the imaging board and stored in a frame buffer. All the operations are performed on this image buffer and the resulting data about the presence of a pothole is sent to the central logic controller.

The series of imaging operations performed on the image buffer is explained below in the control flow diagram shown in Fig. 5:

![Control Flow Diagram](image_url)

**Figure 5. Pothole detection solution methodology.**

First the histogram for the image is plotted to find the image threshold. The histogram graphically displays the distribution of the pixel values in the image. Figure 6 illustrates the histogram for various pothole simulations.
The first maximum in the histogram identifies the pixel value of the background and the second maximum identifies the pothole. This is because the background tends to occupy more space than the pothole or obstacle in any image capture. This is clearly illustrated in all the four cases shown above.

Figure 6. Histograms for different scenarios.

The next step is to choose an appropriate threshold value to demarcate the edges in the image. The second maximum is chosen as the threshold value as this represents the pixel intensity of the portion of the image covering the pothole.

Once the appropriate threshold is set, the edges in the image can be detected. The edges are detected by comparing the values of the neighborhood pixels. Four edge detection operators were compared for the given problem using MathCAD and the Robert’s Edge Detection Operator was selected.

Figure 7. Comparison of Edge Detection Operators
Figure 7 presented above illustrates the performance of different edge detection operators for one particular scenario of the pothole image. The comparisons were also performed for a number of different scenarios and in all the cases, the Robert’s Edge Detection Operator performed better than the other operators.

The next step is to remove noise present in the image buffer due to the lines marking the course boundary and the obstacles. As seen from Figure 6, after the edges are detected there is possibility for a lot of noise to be present in the image. To remove this a simple image scan is done and all the random and linear pixels are removed from the image.

The next step is to find the blobs in the image. The blobs are identified by looking for contours with closed boundaries in the image buffer. There is a possibility for the pothole to occur in the edge of the image, and to accommodate this case the image boundary is also considered as a valid boundary for the contour.

Once the blobs are identified, the size and the centroid of the blob are calculated. If the size of the blob falls within the permissible error range, it can be concluded that a blob is present. The error value is obtained from the calibration procedure, by taking a snapshot of the actual pothole on the field.

### 7. SOFTWARE DESIGN

The software for the solution was designed using a structured methodology. A unique blob analysis algorithm was developed to identify the simulated potholes. The software was initially designed using the Unified Modeling Language. The design was done as a mixture of algorithmic oriented modeling and object oriented modeling to achieve both ease of software creation and ease of software management.

#### Model Description

The model is basically algorithmic oriented with all the sub-systems designed in an object-oriented manner to promote software reusability and management. The whole software was designed as a stand-alone package that can be compiled and linked to the existing control software. The software is implemented in a number of classes and hence data is encapsulated from the main software. The user only needs to create an instance of the `PotResult` class and can get the information about the presence of a pothole by accessing the respective member functions.

The model illustrated in Figure 8 details the construction of the software. Each of the boxes describes a class within the model. The classes in the second tier are the derived classes from the `Pothole` class, and are executed in the order in which they are listed below. All the classes have private data members that hold the image buffer in a 256x256 integer array.
The raw image supplied by the frame grabber is a 256x256 array of pixel values with each of the pixels holding a gray-scale value. A histogram is first developed for the image buffer and a threshold operation is performed to convert the gray scale image into a binary image. An edge detection operation is then performed. At this stage a simulated pothole, if present, would be in the shape of an oblong ellipsoid. There is also a possibility for noise to be present in the image due to various sources – like the lines marking the course or even the obstacles. A noise removal operation is performed to remove noises from the image. A pattern matching logic is used to identify noises from lines and other obstacles.

![Diagram of pothole detection software model](image)

**Figure 8. Pothole detection software model.**

A blob analysis is performed on the image to identify and extract all the blobs within the image buffer. Based on the physical parameters of the camera, an error range has been established for the potential image sizes of these blobs. Each of the blobs identified within the image frame is first checked for the ellipsoid pattern, and the blobs that match this pattern are then filtered using the permissible error values calculated from the physical parameters of the camera.

If there is a blob that falls within the specified error range, then the `PotResult` object is set to a Boolean value of true and the centroid of the blob is also calculated and passed to the `PotResult` object.

### 8. INTEGRATION ISSUES

Integration of the solution with the existing robot systems is very critical to ensure the proper functioning of the robot as a single entity. As the solution for the detection of simulated
potholes was developed with seamless integration in mind, it now becomes easy to integrate it with the other subsystems.

As for the physical location and orientation of the camera, it was mounted on the front top end of the robot to get maximum possible field of view. The software for controlling the hardware was written as a single stand-alone package. Hence the software only needs to be compiled separately and then linked at the appropriate place. Extracting the information from the software also becomes easier as it requires the creation of a single object of the PotResult class, and this object indicates the presence of a pothole and its location.

9. ACKNOWLEDGEMENTS

The authors would like to extend their many thanks to the EPIX, Inc., for providing us their SV4 Imaging Board free of cost. Our thanks are also due to all the sponsors of the Bearcat II, without whose support this work would not have been fruitful. The sponsors are listed at http://www.robotics.uc.edu/sponsors. Special thanks are due to the Robot Vision class students for all their co-operation and enthusiasm.

10. REFERENCES

