AUTONOMOUS INLINE PIPE INSPECTION ROBOT

A dissertation

Presented to

SCHOOL OF MECHANICAL AND MANUFACTURING ENGINEERING

Department of Mechanical Engineering

NUST

ISLAMABAD, PAKISTAN

In Partial Fulfillment

of the Requirements for the Degree of

Bachelors of Mechanical Engineering

By

Syed Furqan Khalil

Usman Khalid

Heena Tariq

June 2017

Keywords: Inline, Pipe, Inspection, Autonomous, Robot

EXAMINATION COMMITTEE

We hereby recommend that the final year project report prepared under our supervision by:

- Syed Furqan Khalil (NUST201305456BSMME11113F)
- Heena Tariq (NUST201305363BSMME11113F)
- Usman Khalid (NUST201304776BSMME11113F).

Titled: "AUTONOMOUS INLINE PIPE INSPECTION ROBOT" be accepted in partial fulfillment of the requirements for the award of B.Sc. Mechanical Engineering degree.

Supervisor: Dr Muhammad Naveed, Asst. Professor Department of Robotics and Intelligent Machine Engineering	Dated:
Committee Member: Dr Yasar Ayaz (HoD), Asst. Professor Department of Robotics and Intelligent Machine Engineering	Dated:
Committee Member: Dr Muhammad Sajid, Asst. Professor Department of Mechanical Engineering	Dated:

_____ _____

(Head of Department) (Date)

COUNTERSIGNED

Dated: _____ _____

(Dean / Principal)

ABSTRACT

The project aims at developing a Pipe Inline Inspection Robot capable of traversing through complex pipe layouts while performing non-destructive testing. This project focuses on a novel mechanism that allows the robot to turn through elbows and fit through a larger diameter range of pipe compared to its previous version, ATOM. Furthermore, the report includes detail of the active diametrical change control module employed and the redesigned passive system conjoined to the mechanism. Finally, AIOM is equipped with a visual inspection module that enables it to process and detect cracks onboard creating a standalone unit that can be used in a variety of scenarios.

PREFACE

This thesis is presented to the NUST School of Mechanical and Manufacturing Engineering (SMME), Islamabad in partial fulfillment of the requirement of the degree BE Mechanical Engineering for the student authors and describes in detail all the efforts that led to completion of their Final Year Project titled “Autonomous Inline Pipe Inspection Robot”. This thesis elaborates extensively all the stages that this project went through from conception phase all the way to the finalization phase and also sheds some light on the methodology and calculations that were adopted in order to design the driving vehicle and the inspection module. While organizing this thesis, care has been taken to strictly keep it in accordance with the recommended format provided by SMME. The authors have made a conscious method to use simple and lucid diction and explain the major concepts of Non-Destructive testing and in pipe locomotion to the readers in a simple yet comprehensive manner. Visual aids like pictures, drawings, tables and graphs etc. have been used wherever necessary to add to the overall clarity of the report. Furthermore, design calculations and formulae have also been written. A dedicated chapter at the end identifies certain areas in which there is some room for improvement to make this machine even better and more useful. This chapter also points out the aspects on which our juniors can work to get better results from this machine.

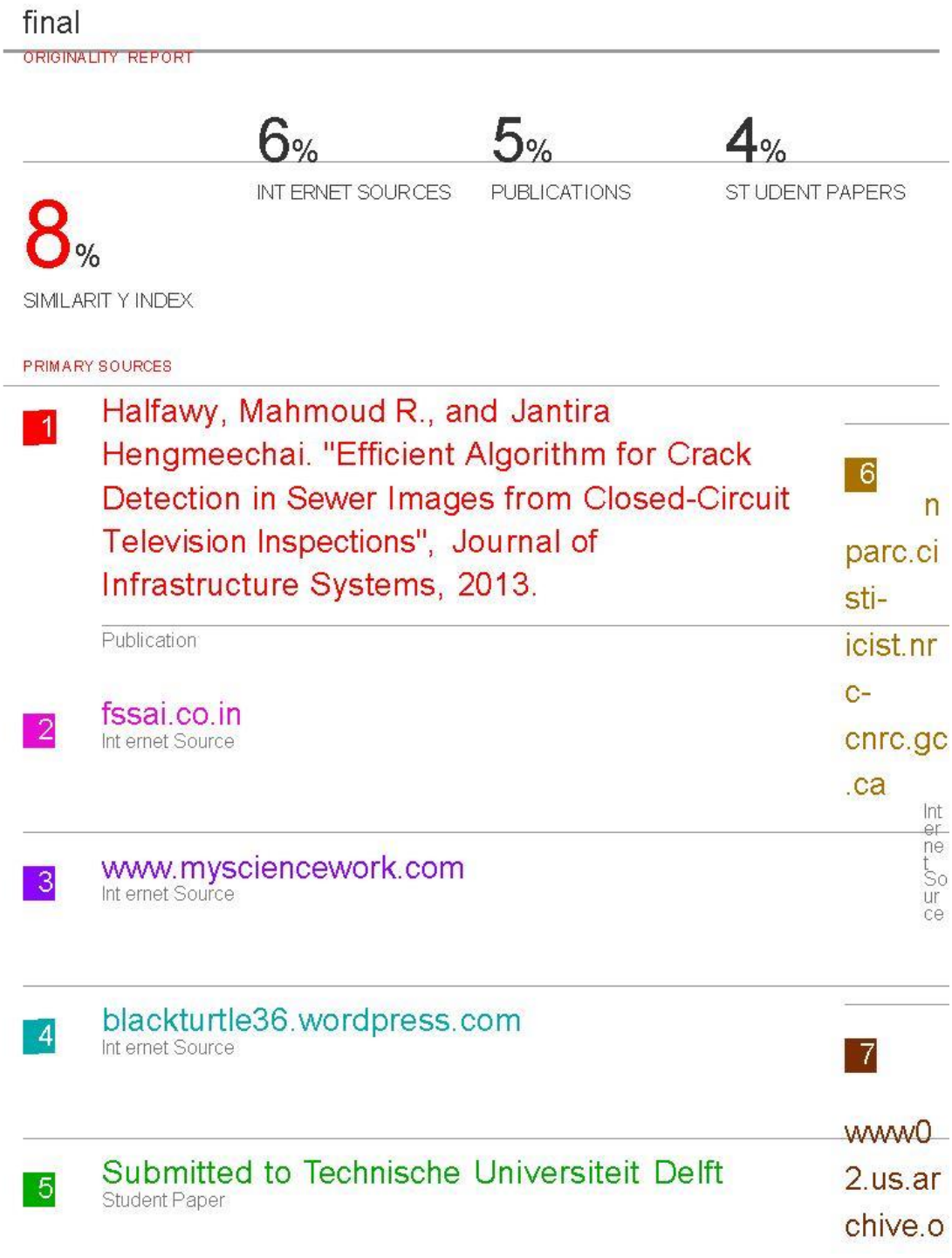
ACKNOWLEDGMENTS

We are highly indebted to our Project Supervisor Dr. Muhammad Naveed & co-supervisor Dr. Yasir Ayaz for their guidance and constant supervision as well as for providing necessary information regarding the project and also for his support in completing the project. They have been a source of continuous support and assisted us wherever possible.

We are also extremely grateful to our seniors Arslan Thaheem and Moiz Ashraf for guiding us through the work they had previously conducted in the project and also for their continued help and support throughout the project's development.

We would also like to pay special thanks to the entire staff of the Ayub Engineering Works who have been really helpful with regards to manufacturing of our project. Special thanks to Mr. Ayub and Mr. Umer for their hard work and dedication. Services of these people are commendable. Without their precious help, we could not have completed our project in time.

ORIGINALITY REPORT



COPYRIGHT

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AIOM (Autonomous Inline Observation Machine)

ROI (Region of Interest)

CCD (Charged Coupled Device)

CCTV (Closed Circuit Television)

MPI (Mean Pixel Intensity)

SID (Standard Intensity Deviation)

ADAMS (Automatic Dynamic Analysis of Mechanical Systems)

CFD (Computational Fluid Dynamics)

CHAPTER 1

INTRODUCTION

Need:

Millions of kilometers of pipelines run across the world. These transfer a wide variety of fluids, most commonly, oil, gas and water. If defects in these pipes are not identified and rectified in a timely manner, vastly disastrous consequences are seen. Two such recent incidents include:

- Dhabeji pipeline burst stopped water to Karachi last month
- Sui gas pipe leak in Mahmoodabad, Karachi

To prevent such disaster and prevent loss to property and life it is necessary to conduct regular inspection. One such mechanism is inline pipe inspection via robot vehicles.

Our project comprises 2 major parts.

- Design, development and control of a robot vehicle that capable of traversing long pipeline structures.
- Study and evaluation of NDT techniques used inside pipelines and then selecting one suited to our needs.

Non-destructive testing for pipeline inspection:

Following are the most widely used inspection methods for detecting surface and sub surfaces defects in pipelines.

1.1.1 Magnetic flux Leakage method:

This method employs a strong magnetic field which is established in the pipe circumference using either magnets or by feeding electrical current to the steel. Defected and cracked regions of the pipe cannot support as much magnetic flux as undamaged areas so magnetic flux leaks out of the pipe wall at the damaged areas. Area of damage is then detected by an array of sensors which then intercepts the magnetic flux.

This method requires an array of large number of sensors around the circumference of the measuring tool to catch leakage from all the sides and accurately measure the defects. Since or designed robot has three legs and is not in contact with the pipe all around, this method cannot be utilized in it. Furthermore, the flux will cause interference with the motor EMF causing mobility problems.

1.1.2 Ultrasonic Testing:

This method is used to accurately measure distance to a surface or thickness of a material. High frequency sound waves are emitted from a transducer into the material and echo signals reflected from inner outer surfaces and cracks of material are received. Time interval between the arrivals of different signals enables the tool to measure width of either crack or whole material.

This method is has very advantages such as

- High penetrating power and sensitivity, allows even very small cracks to be detected.
- Greater accuracy than other NDTs.
- Non-hazardous, portable and highly automated process.
- Quick and easy implementation and results.

This method also requires series of transducers to be fixed all around the circumference of measuring tool (robot) and our design is limited in that sense.

1.1.3 Visual Inspection:

Non-Destructive visual inspection includes using different cameras for video inspection. Some provide specific views of the pipeline while others are used to take high quality images of inside to investigate. Some camera systems allow complete control to aim and zoom thus enhancing field of view and quality.

Inspection camera used can pan and rotate in all direction and is fixed with lights to enhance the visibility. Camera mounted on a tethered or untethered robotic crawler will be sent inside the pipe and inspection can be performed. Such a method is very practical and cost effective for a design such as ours.

Types of Defects:

A wide range of defects and hindrances can occur in pipes which include:

- Corrosion
- Cracks
- Leakage
- Pin holes
- Algae
- Blockage

High resolution Visual inspection will allow many major defects to be detected with ease such as Corrosion, leakage, algae and blockage. These defects are critical to the pipe systems and need to be detected to avoid damage and loss. Furthermore coupling it with image processing unit results in a quicker and more intelligent process.

CHAPTER 2

LITERATURE REVIEW

Roslin et al. [1] in their work have discussed multiple different systems of locomotion for inline pipe robots including hybrid systems incorporating multiple sub systems. These include caterpillar, wheeled and wall press types vehicles. The combination of different forms of locomotion allows more flexible applications. Their work discusses the importance of flexibility in robot vehicles necessary to achieve turning and identifies wall press type robots as ideal for vertical or inclined applications. Within wall press types they discuss different vehicles and conclude that different hybrid systems offer different advantages. The wall press caterpillar robot allows easier navigation through branches while the wall press wheel type gives greater mobility and the wall press screw type performs best in curved pipes. Since our work revolves around the use of a wall press type robot, knowing its implications and advantages becomes important to enhancing its use and utility. None of the models discussed in their work however shows the ability to navigate from bigger pipes to smaller branches.

Nayak et al. [2] have presented their design of a new wall press locomotive systems. Their proposed model is a screw driver adaptable wheeled wall press type robot. The functioning of such a model is fairly different from our standard driven wheeled wall press type but the paper discusses in depth design of such a robot vehicle. They have discussed the main parameters necessary for design and functionality of a wall press robot. These primarily include mass of robot, wheel radius, and static friction and drag forces. The work also studies and presents the minimum torque required to prevent slippage for inclined climb for different coefficients of frictions.

These details help us identify the parameters that need to be taken in to account in designing and re designing a robot vehicle. Before we can move to determining and stabilizing traction forces frictional coefficient between wheel and pipe must be determined from prior data or experimentally.

Choi et al. [3] have worked on the design and implementation of active steering capabilities of in inline pipe robots. They discuss the ability of robots to navigate complex pipe networks that include L and T bends. The set of forces that need to be taken into account while designing the legs of a robot have been described as well as the methods to evaluate the minimum radius an in pipe vehicle is capable of turning given its dimensions and the flexibility of its legs. The model on which they base their analysis is the MRINSPECT IV, a robot capable of passively adapting to the shape and dimensions of a pipe due to a system of links and springs that push the wheels against the wall while at the same time allowing a great degree of flexibility. Their work also demonstrates the capability of the robot vehicle to traverse through bends and pipe fittings while also discussing methods for predicting robot behavior in complex pipe structures.

The MRINSPECT IV provides great insight for developing the passive adaptive module of our own robot vehicle while the methods discussed in the paper help us determine the dimensions of the pipe layout that our existing or modified vehicle will be able to safely traverse.

Roh et al. [9] like Choi [3] base their work upon the MRINSPECT series of inline pipe robots. They discuss both the IV and the III model which are seen to not have very significant differences. They have performed further tests on the model by driving it through a variety of pipe layouts. They further analyze the forces on the moving robot vehicle and discuss its utility in urban pipelines thanks to its 3 dimensional steering

capabilities and outstanding mobility in navigation. Results of preliminary experiments have been used to verify these results.

Further study of the MRINSPECT model allows us to better develop our own vehicle, parts of which are loosely based on the MRINSPECT. The tests performed here display proper techniques of verifying models that we will need to fully or partly incorporate into our work.

Zhang et al. [8] in their work describe active diameter changing of the robot vehicle for it to allow travel in pipe systems of varying diameter. They discuss the parameters including robot geometry, frictional coefficients and inclination angles that must be determined or taken into account to be able to accurately develop a control strategy. Their mechanism involves the use of force detection on the legs to determine when diametric change is necessary. This method also allows the maintaining of a constant traction force between the wheels and walls since traction is a function of the contact force between the wall and the wheel. A constant tractive force ensures that the robot moves smoothly and without slippage.

This study allows us to develop a control system taking sensory information regarding the contact force on the legs of the robot and outputting a signal that either retracts or expands the leg. We thus develop a system capable of maintaining a constant traction force and also therefore of altering diameter as necessary.

Song et al. [11] in their work discuss the kinematic properties of a wheeled mobile robot in a cylindrical workspace, that is, a pipe. They start with discussing the kinematic properties of a single wheel and then analyze the geometric constraints of a wheeled robot in pipe with analytical geometry. Based on these analyses they discuss kinematic

properties and then simulate the rolling of a single wheel to verify results. While they primarily focus on car type robots which would have different geometric constraints and wheel orientation as compared to our wall press type robot, we can draw upon their analyses and results to develop our own kinematic model.

Zhao et al. [12] have developed a kinematic model of a wall press type robot and simulated its motion in a straight round pipe using MSC ADAMS. Their model resembles the MRINSPECT IV we have previously seen adopting a similar mechanism. In their work they discuss the theoretical background of the modeling and provide in depth detail of the modeling process and their results. As we work on modifying our existing robot to be able to navigate an elbow it would be constructive to simulate its potential motion in a similar manner. Where Zhao et al. have used a straight pipe we can develop a curved one.

Alexander Reiss [4] has presented the numerous forms of NDT that may be used in pipe inspection, the most prominent of which are Ultrasonic, Laser profilometry, radiography and visual inspection. Ultrasonic inspection is generally reserved for flooded or partially flooded pipes as ultrasound waves require a medium of travel. In unflooded pipes a modified probe may be used but that requires constant contact with the pipe surface. Laser profilometry coupled with a CCD camera, while extremely suitable for detection of surface defects is a highly expensive form of NDT primarily because of the stringent accuracies involved. Film based radiography is an effective technique for locating and evaluating discontinuities. However due to the limitations of the amount of area that can be inspected in a single go, radiography is generally reserved for weld inspection which is a well-defined specific region. Visual inspection remains the preferred mode of inline pipe inspection throughout the world. It generally involves an operator viewing the video feed in real time which can take up to several hours for longer pipes.

For an operator to view possibly hundreds hours of video makes the process time and labor intensive as well as opens the door for human error. One way around this is on board or remote image processing for defect identification. **Safizadeh et al.** [5] study the utility and methods of image processing in their work. It should be noted that image processing here includes both laser profile images on a CCD camera and standard RGB video from a CCTV camera. In their work Safizadeh et al. focus mostly on processing images obtained from a circular ring shaped pattern projected by a laser diode. They describe a novel method for extracting and analyzing intensity variations in the obtained images and consequently producing an accurate representative image of the pipe wall.

While we are not inclined to use laser profilometry in our work, it can be observed from their work that image processing can come in extremely handy in defect analysis greatly reducing required time and effort.

Halfawy et al. [6] and **Rzhanov** [7] in their respective works discuss 2 alternative ways of processing CCTV images to achieve the desired results. Rzhanov suggests a photo-mosaicking approach to combining images to create a single 2D picture easier to view, store and analyze. This method, while it reduces the chance for error and allows more intricate observation maintains the disadvantage of being labor intensive and time consuming. Halfawy however suggests an efficient algorithm for computing the presence of a crack in an image, thereby allowing the machine itself to determine and record the presence of a defect.

Halfawy et al. [6] first identify the visible features that are generally common to cracks and exploit these visual characteristics to efficiently identify actual cracks and filter out

background noise. Their algorithm consists of three main steps. The first is the preparation of the CCTV image for crack detection by identifying a set of candidate crack fragments using the Sobel method to detect horizontal and vertical edges separately. The second step enhances candidate crack segments by filling the gaps between closely adjacent and aligned edges and merges fragments that potentially represent the same crack. In the third step two filters are defined and applied based on prior knowledge of visual characteristics of cracks to remove noise. Extensive testing is performed to prove the robustness of the algorithm.

We base our own work loosely upon that of Halfawy, using similar criteria for crack identification and adopting many of the tools that they have used in their work. Their algorithm while exceptionally accurate as shown by their test results has the limitation of being computationally demanding which is why they limited their work to analyzing existing CCTV footage as opposed to conducting a real time analysis. We will need to enhance algorithm speed before we can effectively process images in real time.

Similar to Halfawy, Iyer et al. [10] present another algorithm for detecting cracks inside pipes. They however use slightly different visual criteria for identifying a crack. They define cracks in pipe images as clearly visible patterns (darkest in the image), locally linear and branching in a piece-wise fashion. Like the previous work, here cracks are first enhanced by mathematical morphology with respect to their spatial properties. In order to differentiate cracks from analogous background patterns, cross curvature evaluation followed by linear filtering is performed. Extensive testing is performed to check the robustness of the algorithm.

The contrasting yet reasonably accurate approaches used in this work and the previous show that different method may be used to identify and isolate cracks. The first steps however are properly defining what features comprise a crack. In both cases however the

analysis was limited to still images or previously recorded video. To process images in real time a faster approach would be needed.

CHAPTER 3

METHODOLOGY

3.1 Design Elements

3.1.1 Initial Design:

The basic design of our inline inspection (ILI) robot consists of three legs 120^{o} apart which can adjust their length according to pipe diameter within certain limit. At their roots, these legs are collectively jointed to a screw which is rotated by a stepper motor allowing adjustments. The scissor mechanism implemented allows variation in leg length which is the most significant advantage of this ILI robot over other such as pigs. Components of design are as follows:

1. **Servo Motor:** It controls the motion of wheels making the robot move forward and backward.

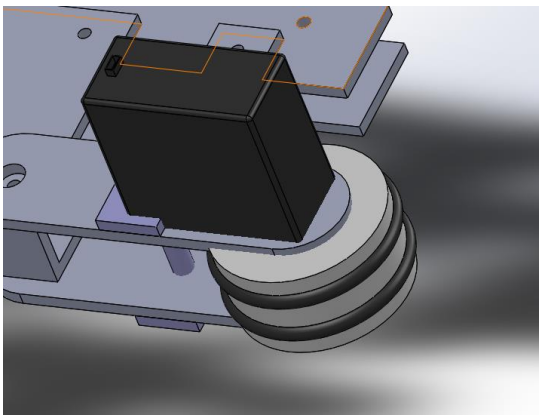


Figure 1 Servo motor attached to one wheel

2. **Stepper Motor:** It controls the length of legs making the robot to adjust to different diameters with the help of screw action.

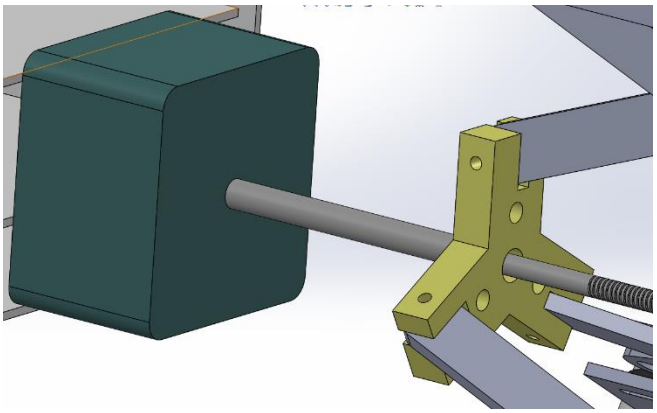


Figure 2 Stepper Motor placed on Lead Screw

3. **Main Lead Screw:** At the center of whole body, connects legs to stepper motor and allows diameter variation.

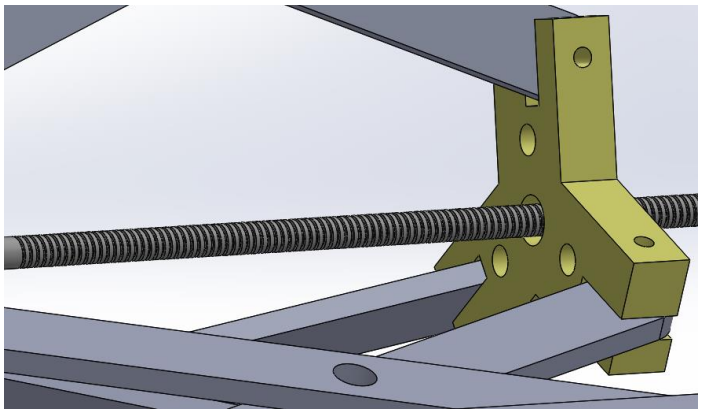


Figure 3 Centrally Placed Lead Screw

4. **Scissor Mechanism:** Scissor mechanism is fitted on screw which when rotates, causes extension and compression of scissors.

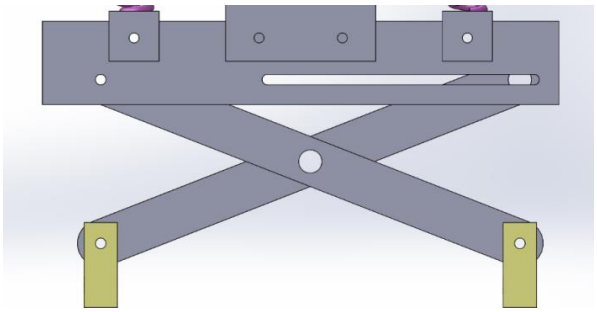


Figure 4 Scissor Link Mechanism Employed

3.1.2 New Design:

The previous design of AIOM had many limitations owing to certain constraints that rendered the robot's mobility difficult in complex pipe structures such as bends and elbows. Moreover, the system was unable to provide adequate tractive force for smooth movement through the pipe while movement through a vertical pipe structure was not possible.

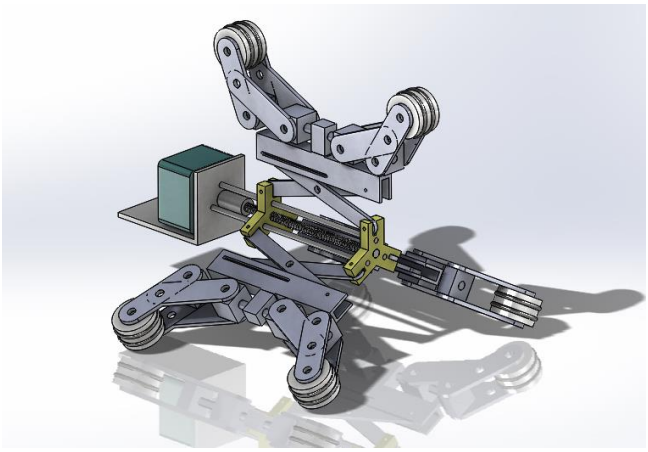


Figure 5 SolidWorks Model of New Design

To overcome all these problems and to make a design that gave enough tractive force for vertical climb alongside flexibility for movement through an elbow, pantograph linkage system was employed. Key features of this system are as follows:

Clearance:

Redesigned legs have ample amount of clearance in comparison with the old one which enable the robot to move through bends and elbows. It also increased the passive flexibility of robot so it can adjust to small bumps on the inner pipe surface. Furthermore, the new design places the two wheels at a greater distance which increases the grip area. This in turn ensures that the robot's motion is more stable.

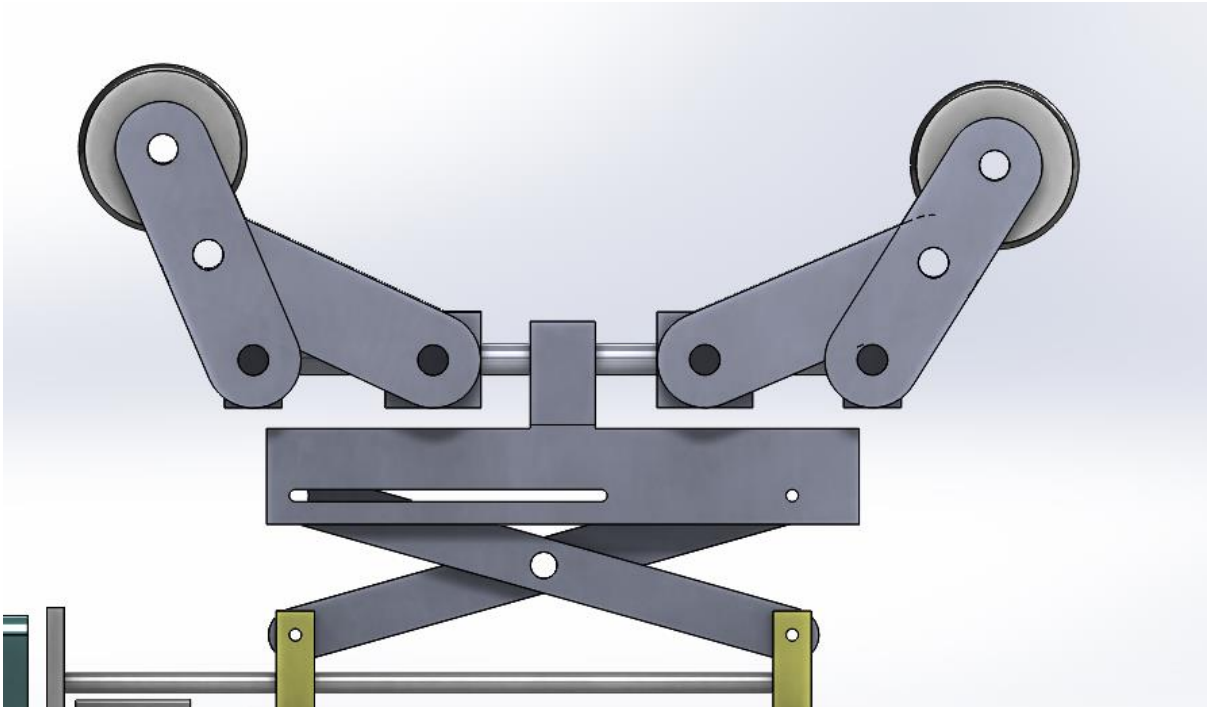


Figure 6: Passive Diametrical Module

Slider Roller Bearing:

New system has slider roller bearing for flexible spring action offering almost no resistance to passive adjustment. This eliminates the possibility of robot being stuck in the pipe due to excessive resistance that previously caused problems. The straight cylinder on which slider bearing is mounted made it feasible to measure normal force on each of the leg. A slider potentiometer moving along the linear motion of bearing gave corresponding voltage depending upon its position to measure how much force is being imparted on to the wheels.

Figure 7: Slider Roller Bearing

Supported springs:

Springs in this mechanism are mounted on a cylinder which prevents them from buckling when compressed under the action of normal force; this ensures a fluid compression throughout the spring range without resistance. The spring stiffness is calculated based on normal force that is required for normal and as well as vertical movement.

3.2 Navigating In Elbow:

AIOM was designed to be able to navigate through pipe elbows. The robot for now however is incapable of detecting bends ahead of it. This may be achieved in the future using multiple ultrasound sensors to determine both the distance and direction of the bend. For our purposes however we use the curvature of the bend and the orientation of the wheels to determine the velocities that need be applied to each driven wheel to achieve smooth turning. We then experimentally validate the results by turning the robot through the bend.

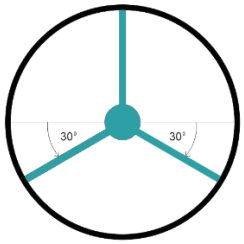


Figure 8 Distribution of Legs in Robot

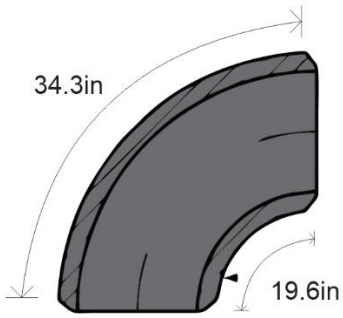


Figure 9 Pipe Layout & Dimensions

The outer and inner arc lengths of the pipe are measured as 34.25'' and 19.625'' respectively. We evaluate the actual distance travelled by each of the 2 legs at 30^{o} to the horizontal. We assume the change in length from the shorter to the longer as uniform along the circumference. That is the change in length per degree is:

$$\frac{34.25 - 19.625}{180}$$

For 30^{o} actual travel length becomes 22” for the shorter arc and 32” for the longer arc. The ratio of speeds that must be assigned to each of the two legs is therefore 1:1.6. The top leg lies effectively midway between the left and right leg and may therefore be assigned a speed that is the average of the 2 speeds.

3.3 Adaptive Diameter Mechanism and Tractive force regulation

3.3.1 Tractive Force

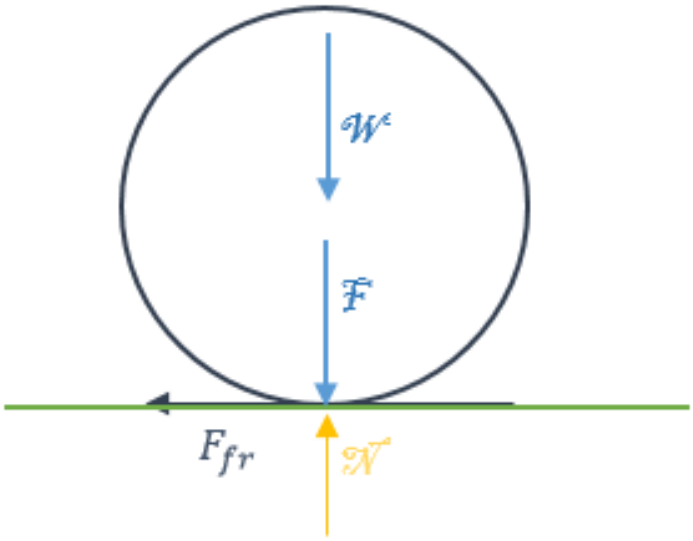


Figure 10 Free Body Diagram of Wheel

The above figure shows the forces acting on a single wheel. Depending on the orientation of the wheel W may act towards or away from the pipe wall. In case W acts away from the pipe wall, $N = F - W$, where N is the normal contact force and F is the pressing force generated by the pressing of the wheel against the wall due to the diameter adaptive

mechanism. The tractive force generated along the wall largely depends on the frictional force between the wheel and the wall and may be determined by:

1. Normal Contact Force
2. Coefficient of friction

$$N = F + W$$

OR $N = F - W$ (for top wheel)

$$F_{fr} = \mu N$$

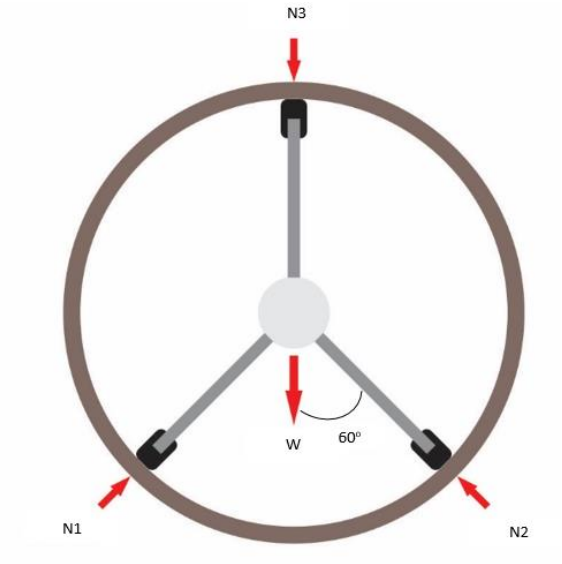


Figure 11 Contact Forces on Robot

It must be noted that the speed of the robot is not merely a function of motor speed but a function of tractive force. It is thereby necessary to determine said force to ensure a constant speed as necessary. The tractive force in each wheel is dependent on the contact force on the wheel and the coefficient of friction between the wheel and the pipe wall. The total tractive force is the sum total of the forces on the 3 wheels.

We do a static analysis of the above body to determine relation between the above forces for optimal tractive force. It should be noted that the our diameter adaptive mecanism allows control over the 3 contact forces by means of extension or retraction of the leg. This contact force is measured by means of a force sensitive rensistor mounted below the wheel.

It should be noted that while N_3 may appear to be 0 that is an undesirable case. To ensure tractive force is produced in each of the wheels and to maintain a certain degree of stability it is necessary that the top wheel be pressed against the wall thereby producing a certain contact force. The relation between the top and bottom contact forces for optimal running needs to be determined. This relation will allow us to control the adaptive mechanism via a single force sensor as the other legs may be adjusted before hand to ensure a certain force ratio.

$$\Sigma F_x = N_1 cos60 - N_2 cos60 = 0$$

$$N_1 cos60 = N_2 cos60$$

$$N_1 = N_2$$

$$\Sigma F_y = N_1 sin60 + N_2 sin60 - W - N_3 = 0$$

$$2N_1 sin60 - W - N_3 = 0$$

$$\boldsymbol{1.73N_1 = N_3 + W}$$

This final equation gives us the relation between the 2 forces that needs to be maintained. Since the Weight, W of the robot remains constant for a given specification, the desired ratio may be maintained between the three contact forces by adjusting default leg length or the spring constant in use. This therefore allows the control of all 3 legs by a single

mechanism. The current weight of the complete vehicle is approximately 60 N, but that may vary as additional components such as sensors are installed.

3.3.2 Importance of tractive forces:

As a wall pressed type robot, tractive force plays a crucial part in the motion of AIOM. Tractive force is the force required to produce smooth motion between a body (wheels) and a tangential surface (pipe inline) through the use of dry friction. It decides the maximum speed of the robot for a particular power input.

As the robot relies on the constant contact between its wheels and the pipe to travel through pipes and complex layouts such as elbows, it is necessary for this tractive force to remain at an optimum value that ensures that the robot is able to move within varying sized pipes, while maintaining a constant servo motor output.

The tractive force present decides the power required by the servo motor to drive the wheels such that the robot travels at an optimum speed that allows it to efficiently visually inspect the pipe inline and detect cracks. This power requirement furthermore also decides the battery capacity needed for the robot to travel a particular distance. A larger tractive force subsequently means a bigger battery needs to be placed on the robot which increases the weight of the robot and additionally hinders smooth motion of the robot.

Thus, it is in our best interest to ensure that tractive forces felt by robot stay constant. As will be seen in the following text, a major portion of our project was dedicated to devising a self-actuating mechanism solely aimed at maintaining previously mentioned tractive forces constant.

3.3.3 Traction force optimization:

The optimum traction force available is dependent on motor torque and the wheel radius:

$$F_{tr} = \frac{\tau}{r}$$

Based on our servo motor data sheet [13] the maximum torque produced by the motor is 38 oz-in or 0.27 Nm. The design radius of the wheel is 1.5 cm. Optimum tractive effort is therefore:

$$F_{tr} = \frac{0.27}{0.015} = 18\ N$$

The optimum tractive effort that we have evaluated is equivalent to the maximum tractive effort that may be produced by the motor. By maximising the tractive effort we significantly reduce wheel slippage. Since our primary method of localization is wheel odometry; to achieve reasonably accurate results it is necessary that slippage be kept as minimal as possible.

Tractive effort like friction is a function of the frictional coefficient and contact force acting on the wheels.

$$F_{tr} = c_{fr} \times N$$

While the coefficient for any particular pipe material would remain constant we are able to control the tractive effort based on the compression of the spring in each of the driven wheels. The spring when compressed exerts a force on the wheel pushing it against the wall thereby increasing the contact force. To determine the optimal level of compression we first conduct an external static analysis of the robot. (see fig. 3.2)

For Nylon on PVC we obtain the frictional coefficient from [14] as 0.3.

Required contact force is therefore:

$$N = \frac{18}{0.3} = 60N$$

Robot weight = $5.1kg \times g = 50N$

Angle between weight and leg = 60^o

$$N_w = 25\cos(60) = 12.5N$$

$$N = N_w + N_s$$

$$N_s = 47.5\ N$$

Spring constant, k = 35 N/cm as determined experimentally.

$$N_s = kx$$

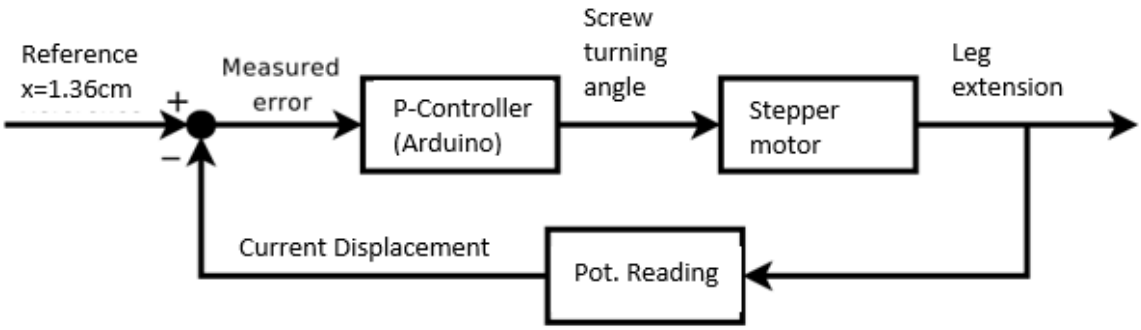
$$x = 1.36\ cm$$

The compression of the spring needed to achieve optimal tractive force is therefore 1.36 cm. A higher compression would drain excess power from the motor and reduce mobility while a lower compression would enhance the chances of wheel slippage.

To maintain this specific level of compression we design a proportional controller using a linear potentiometer that takes the input as the current spring compression and outputs a signal to the central stepper to either expand or retract the robot legs till optimum compression is achieved.

3.4 Control Mechanism:

While we would ideally like to measure the actual force acting on the wheel, it is much easier to measure the compression of a spring, the ideal value of which we calculated above. This measurement is achived simply by connecting a linear potentiometer to the collar dirving the spring. The adaptive mechanism consists of a single lead screw conncected to a stepper motor which when turned causes extension or contraction of the legs thereby altering the diameter of the machine and varying the normal and therefore tractive force. We have designed our controller based on [8] albeit a simpler version.



The reference displacement mentioned above represents the contact force needed to achieve the desired traction as calculated in the previous section. We use a P controller as opposed to a PID controller since the computational cost outweighs the small increase in accuracy and signal smoothing. It should be noted that diametral change that occurs in a nozzle or reducer is in itself a gradual process and therfore renders the need of a higher level of control that is PID extremely unnecessary. The controller then drives the motor that turns the screw to produce the change necessary to achieve the desired traction force or diameter change. The potentiometer then provides a new force reading.

3.5 Visual Inspection NDT

The various forms of NDT commonly used were discussed earlier. The 3 possible methods that we shortlisted and believe capable of properly conducting internal pipe NDT:

1. Ultra sound
2. Laser Profilometry
3. Visual

We were forced to discard Ultrasaound due to the limitation of its ability to only function inside flooded pipelines, while our robot primarily functions in drained pipes. It was observed in [5] that laser profilometry carries the greatest potential for detecting surface

defects, the costs associated do not allow us to use this method. We are therefore left with visual inspection.

Most visual NDT techniques in use today involve an operator guiding a robot vehicle while viewing the video feed. This can possibly take 10s of hours for several km in length of pipes. We therefore intend to incorporate a computer vision technique that will be able to detect the presence of defects and record their existing locations. This information may then be processed to create a layout of the entire pipe network including the location and orientation of the defects. Since the operator now has to review a significantly smaller amount of information, the chance for human error is significantly reduced.

3.5.1 Crack Characteristics:

Before we discuss techniques capable of identifying cracks in an image it is important to classify what may be qualified as a crack. We therefore list individualities generally unique to cracks. Cracks may be characterized as:

1. Darker than most other features
2. Long
3. Irregularly linear (some deviation from a straight line)

It should be noted that all cracks may not conform to this generalization, but these underlying assumptions let us take the first steps to developing a robust algorithm.

3.5.2 Canny Edge Detector:

Our proposed crack detection algorithm is based on the Canny Edge Detector developed by John F. Canny in 1986. The Canny operator was designed to be an optimal edge detector taking a monochrome image as input and producing a binary output image showing the positions of tracked image discontinuities. While in our work we do not alter the internal code of the Canny Edge Detector available with OpenCV packages, an

appreciation and understanding of its working is necessary for the additional steps to be taken. The edge detector is a multi-stage algorithm. The first step is smoothing the monochrome image by applying a Gaussian filter thus reducing noise. This is followed by evaluating intensity gradient of the image by a Sobel filter. The intensity gradient is basically the difference in intensity of 2 neighboring pixels. The gradient of each pixel is evaluated in both the horizontal(x) and vertical(y) directions and the edge gradient and orientation evaluated as:

$$Edge\ Gradient = \sqrt{{G_x}^2 + {G_y}^2}$$

$$Angle = tan^{-1}(\frac{G_y}{G_x})$$

This is followed by a non-maximum suppression that reduces all pixels that may not be an edge to zero. This is achieved by scanning the image pixel by pixel and identifying the local maxima. The last step, hysteresis thresholding is of particular importance in our work because it is elements of this step that we will control for our own algorithm. For this final step Canny uses an upper and a lower threshold. The intensity gradient of all pixels that were previously identified as potential edges is checked against the thresholds. All pixels with an intensity gradient above the Upper Threshold are identified as an edge while all pixels below the Lower threshold are suppressed to zero. A pixel with an intensity gradient between these two values is identified as an edge only if it is connected to a pixel that lies above the Upper threshold; otherwise it is suppressed to zero.

3.5.3 Setting Canny Thresholds:

In most applications a certain set value may be used for the upper and lower thresholds of the Canny operator. However, where images are expected to vary as is in our case, fixed thresholds would not yield sufficiently accurate results. The inside visual environment of a pipe may vary significantly due to rusting, sedimentation, algae formation and pipe

material. The identification of cracks on such varied backgrounds requires that a dynamic threshold be used for edge detection. Here we propose two image parameters that may be used in setting the upper and lower thresholds.

1. Mean pixel intensity: The mean pixel intensity of an image is a single integer that is the average intensity of all the pixels in the image. For an 8-bit monochrome image this number would range from 0-255. Assuming that the crack in the image covers only a minor portion of the entire image, the MPI is essentially a representation of the background on which the crack (if present) forms. A larger MPI therefore denotes a lighter (whiter) background and thus a more pronounced crack (essentially a collection of black pixels). For a higher MPI we can therefore safely set a higher upper threshold for the Canny operator and still expect to sufficiently capture the crack as an edge. This allows us to filter out a number of other features that may otherwise be wrongly identified as cracks.
2. Standard Intensity Deviation: The Standard deviation of pixel intensity in monochrome image denotes the average deviation of pixel intensity from the mean intensity. Depending on the image this value may be as small as 2 or as high as 100, but in most of the cases we have dealt with, the SID was observed to lie between 10 and 60. Again assuming that the crack itself would form a minority of the image, the SID would be a representation of the variations in the background. A higher SID and therefore a higher degree of variation would thus mean that there are a greater number of features that may falsely be identified as cracks. It should be noted that cracks tend to form the darkest parts of an image. Other features such as patches of rust or scratches are generally lighter. Therefore as the lower threshold of the Canny operator is increased, the ‘extras’ get filtered out first. The SID therefore assists in setting the lower threshold.

Based on the 2 parameters (MPI and SID) defined above we can now set dynamic thresholds for the Canny operator depending on the image obtained. As mentioned earlier

the MPI may vary from 0-255 while the SID in general varies from 10-60. These values differ from the range of the Canny operator threshold that is 0-1000. A scaling factor is therefore necessary to determine the actual threshold values to be given to the Canny operator.

3.5.4 Determining Scaling factor:

The scaling factors for the MPI, to set Upper Threshold, and for the SID, to set Lower Threshold, were determined empirically using a library of 40 images (20 cracked, 20 un-cracked). The scaling factor providing the most accurate results was deemed appropriate. Accuracy is defined as the percentage of images correctly identified as positives for cracks. A coarse iterative approach (increments of 1) was used to obtain a rough value for each of the scaling factors. This was then followed by 2 further finer iterative methods (increments of 0.1 and 0.01 respectively) each time narrowing the range of the SF. The final values for the scaling factors were determined as 3.82 for MPI and 14.4 for SID which presented an accuracy of 95% (after incorporating other steps of the algorithm).

3.5.5 Extracting Cracks from Detected Edges:

The steps described above lead up to the Canny Edge Detector that outputs a binary image where each pixel is identified as part of an edge or zero. Dynamic thresholding of the Canny operator does not filter out all of the edges that may wrongly be identified as cracks. We therefore rely on other predominant characteristics of cracks to segregate identified edges into cracks and non-cracks. As mentioned earlier cracks are long and irregularly linear. Based on these 2 underlying assumptions we shall conduct further analysis.

As mentioned beforehand, the Canny Edge Detector outputs a binary image identifying the detected edges. This image however forms no correlation between the pixels identified as edges. We therefore introduce another function from the OpenCV library,

Find Contours. This function takes as input a binary image and outputs an array of contours where a contour is a collection of pixels directly connected to each other. Each contour represents a separate edge. Said edges include cracks as well as other features. It should be noted that this method may not take single cracks as whole. Discontinuities may arise in edge formation due to lighting or texture and single cracks may appear broken up into many. This however does not interfere significantly with our next steps.

To isolate cracks we first determine the length. The length of each contour may simply be approximated as the number of pixels in the contour. For our purposes we have set 30 as the minimum number of pixels that a contour must contain for it to be considered a potential crack. This limit does impose a certain restriction on our method rendering it incapable of identifying smaller cracks.

The previous steps give us a list of potential contours positive for cracks. Each contour is then approximated as a 1^{st} degree polynomial that is, a straight line. The error about this linear approximation gives an estimate of the contour's deviation from a perfectly straight line. The total error is normalized by dividing by the length of the contour to obtain the error per unit length or error per pixel. Cracks tend to be less linear then scratches or joints but more linear then patches of corrosion or algae. An upper and lower bound for normalized error may thus be set to identify cracks. These bounds were determined empirically using our library of images. Suitable values for the upper and lower bound were found to be 0.05 and 0.4 respectively.

Figure 12 Actual Grayscale Image

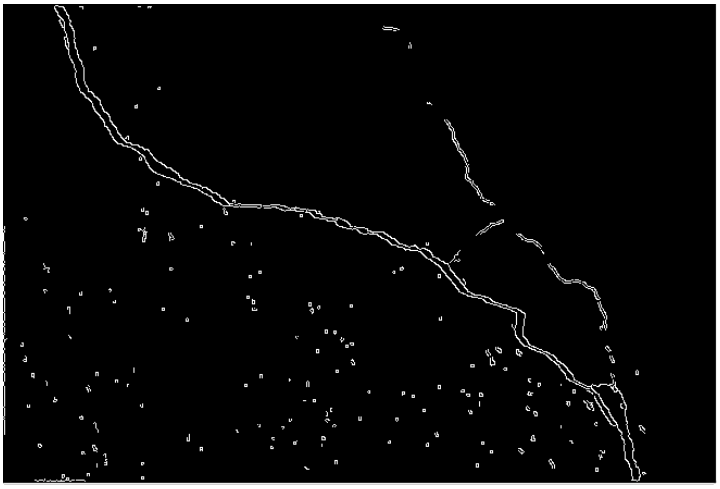


Figure 13: Binary picture after Canny Operator. A large number of ‘extra’ edges are visible

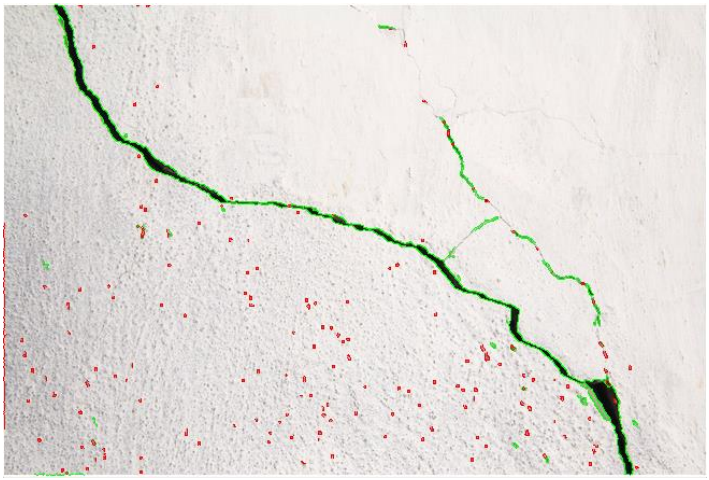


Figure 14 Image after checking for crack parameters. The parts identified as cracks are highlighted in green while the edges not identified as cracks are highlighted in red.

3.5.6 Sensing Hardware:

The hardware used is a Raspberry Pi 2.0 board and a Raspberry Pi Camera Module V2 which has a Sony IMX219 8 Megapixel RGB sensor. The camera module is mounted with a wide angle lens providing a 120^{o} field of vision. The frame rate is set to 3 frames per second. While this may seem low, it is sufficient to capture full cracks without interruption although it does limit the maximum speed the robot may travel at. It should also be noted that the Raspberry Pi 2.0 is a mid-range board and higher powered processors may allow for sufficiently improved functionality.

3.5.7 In pipe crack identification:

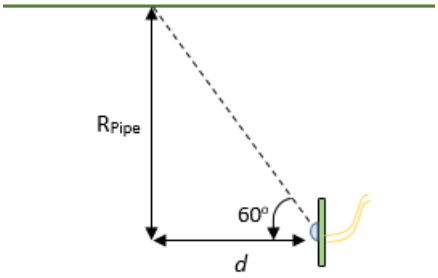
For detection and analysis of cracks inside pipes we modify, or rather extend the method proposed earlier. As mentioned above the camera module obtains a series of images (at a

rate of 3 frames per second). Each image undergoes the steps detailed in the previous sections and edges identified as cracks are obtained. A camera inside a pipe is able to obtain an image several feet into the pipe with decreasing view of the wall the further into the pipe the image goes. For our purpose we require only a certain region that is visible with adequate clarity. We therefore define a region of interest bounded between two concentric circles. Cracks that fall within this region are recorded while the remainder of the image is ignored.

3.5.7.1 Defining the region of interest:

We define the region of interest with two concentric rings that enclose a part of an image a set distance from the camera. This distance (d) is calculated using the pipe radius and the angle of view (φ) which is 120^o in our case:

$$d = \frac{R_{pipe}}{\tan(\frac{1}{2}\varphi)}$$



Given the resolution of the image we set the diameter of the outer ring equivalent to the height (or width if that is the smaller dimension). The inner ring we have set to be 0.85 times the height. This value has been set taking into account the frame rate and the speed of the robot vehicle. The closer this ratio is to 1 the finer our region of interest would be and the more accurate our analysis. However this would require either the robot to travel at a slower speed or a higher frame rate and therefore greater processing capabilities.

Figure 15 Region of Interest of the robot is captured between two circular rings

3.5.8 Attaining relevant data:

Now that we have processed the image and defined the region of interest within the image, we move to extract information that will be used for further analysis. Our main goal here is to be able to identify the location and orientation of a crack in a pipe. For this we record two parameters whenever a partial crack is identified in the defined region of interest. The first is the distance(s) that is the sum of the distance the robot has travelled since the start of its journey and the constant distance (d) between the camera and the ROI that we calculated earlier (See fig.). We employ wheel odometry using motor

encoders built into the driving motors to determine the total travelled distance.

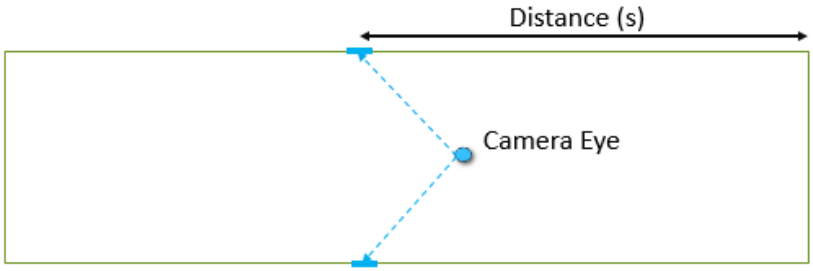


Figure 16 Length parameter used in localization

Admittedly this is a crude method of localization of the robot and could significantly be improved if used in conjunction with other methods like the particle filter.

The second parameter of significance is the orientation of the partial crack within the region of interest (see fig.). This is determined by calculating the angle (Θ) of the crack with respect to the horizontal axis. For this we first determine the center point of the contour(s) that form the crack. This point will be considered representative of the entire crack. Using the x and y coordinates of the center we evaluate the angle simply by taking the arc tan of the ratio of the y and x distances from the origin that is the center of the image.

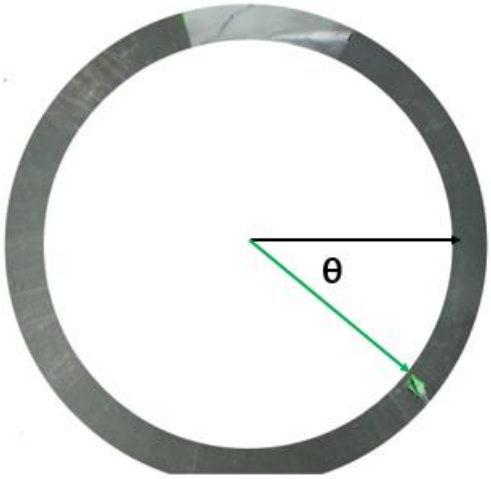


Figure 17 Angle parameter used in localization

3.5.9 Visualization of results:

In this way the robot processes images on board and records a minimal amount of information. This information is stored as a list of coordinates (s, Θ) that is processed

upon completion of the robot's journey. Another parameter that is recorded although not by the inspection module is the radius of the pipe. In the previous sections we discussed the robot's ability to adapt to changes in pipe diameter based on the compression of the wheel. The sensory information regarding the wheel's compression and the state of the scissor mechanism (and therefore the stepper motor) provides sufficient information as to the radius of the pipe. With these 3 parameters, the post processor is able to recreate a 3D image of the pipe identifying the regions of defect along its length. See fig.

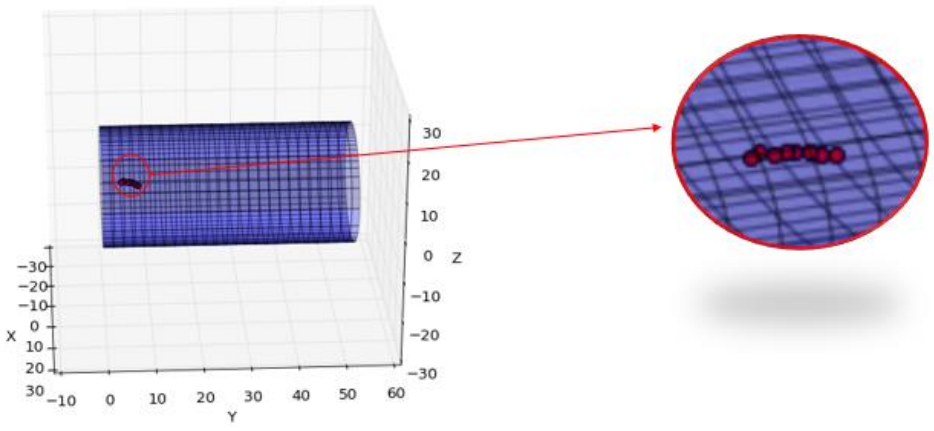


Figure 18 Final Results Obtained For Visualization

CHAPTER 4

RESULTS

AIOM is a mechanically manufactured inline inspection robot fully capable of:

1. Linear motion in pipes ranging from 12-18 inches.
2. Smoothly turning in elbow configurations
3. Performing Visual Inspection and crack detection in real-time.

By redesigning the passive diametrical module of the robot, we have successfully managed to increase the flexibility and maneuverability of the robot that allows it turn through curves and bends in addition to travelling linearly with minimum vibrations, an important criterion for visual inspection to be carried out well.

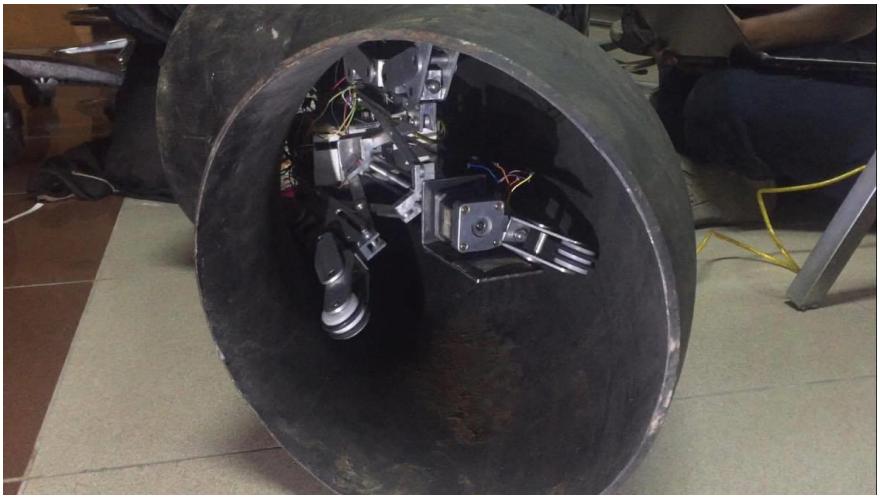


Figure 19 AIOM maneuvering a bend

Furthermore, by successfully incorporating a slider potentiometer as a feedback device, we have enabled the robot to actively resize itself to an optimum size to maintain a constant tractive force irrespective of the pipe diameter. This has rendered the robot capable of vertical motion inside pipes.

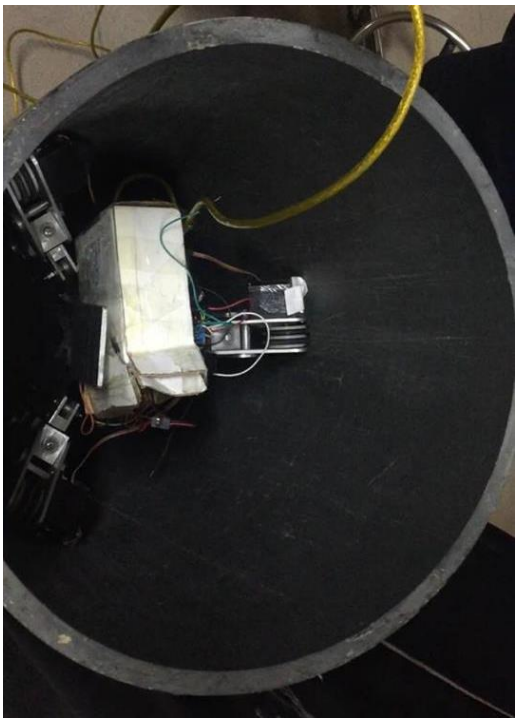


Figure 20 Robot moving vertically in pipe

Finally, AIOM can effectively carry our real-time crack detection using the Raspberry Pi 2 and its camera. The algorithm employed detects cracks with an accuracy of 95%.

The visual inspection module placed at AIOM's front end enables it to constantly monitor the pipe inline throughout the duration of the travel and record the location of each crack detected. This data can then be translated into a plot that pinpoint the cracks position and eliminates the need of an operator to monitor the video feed.

Finally, to validate our results a model was created on ADAMS (*Automated Dynamic Analysis of Mechanical Systems*) to perform a multibody dynamics simulation. The aim of the analysis was to support the new relationship generated between the wheel and the

subsequent motion of the spring.

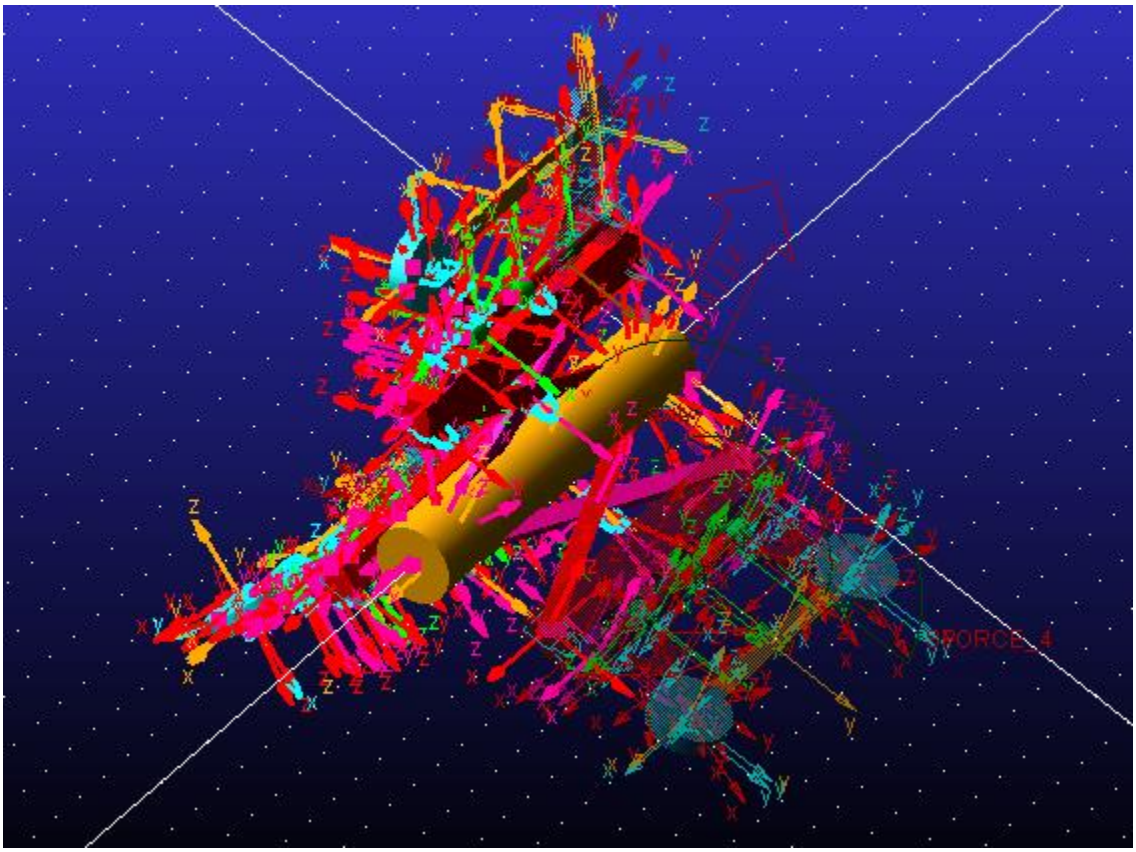


Figure 21 ADAMS Simulation displaying frames

Thus, we can conclusively say that the new design implemented is far superior as it allows greater flexibility, higher maneuverability while maintaining excellent grip/ tractive force with the wall allowing the robot to move smoothly through complex pipe layout.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

As per our initial aims and objectives, we were successfully able to upgrade the previous version of the Inline Pipe Inspection Robot developed at SMME NUST. The first of our 3 main objectives was modifying the link structure of the robot to be enhance stability and functionality. Our final design, we validated by conducting a kinematic simulation on MSC ADAMS where we were able to judge the motion of the individual links based on an applied force and thereby ascertain whether such a structure could allow the robot to turn through a bend. After the manufacturing of the new mechanism, we tested the motion of the robot through a 12" elbow. We observed that it was needed to modify the speed of each of the three driven wheels based on their respective distances to allow the robot to turn smoothly. While the robot was successfully able to navigate through pipe elbows, for now it is incapable of detecting bends ahead of it. This may be achieved in the future using multiple ultrasound sensors to determine both the distance and direction of the bend.

Our next objective was developing a feedback control system for the robot to allow active diameter adaptability based on the force acting on the wheels. While we had initially planned to use Force Sensitive Resistors, we later observed that measuring the spring compression instead of the actual force was just an effective a sensory tool while being more convenient and robust. This was done by using a linear potentiometer to determine the lateral displacement of the collar connected to the spring. The resultant reading would result in actuation of the stepper in either direction. While the control system functioned seamlessly it was observed that the stepper would often stall when required to push the wheels against the wall. In the future to achieve better functionality it would be more appropriate to use a more powerful or multiple steppers.

Lastly, was incorporating an effective NDT technique. The robot was fitted with a visual inspection module and we tested out a computer vision algorithm aimed at being able to detect, and record cracks on the inside of a pipe. While we observed that there were limitations of using a standard RGB camera for instance the inability to detect sub-surface cracks, the use of image processing techniques allowed us to enhance its utility to some degree. By processing images on-board, the amount of information needed to be stored was significantly reduced while post processing of said information allowed detailed maps of pipe and defects to be generated thus reducing the need for an operator. Due to lack of availability of testing pipes, we were not able to perform as extensive testing as we would have liked. In the future, the computer vision technique could be extended to detect defects other than just cracks such as patches of corrosion or algae deposits.

The project possesses still a great potential for improvements. One of the biggest drawbacks of AIOM is its inability to operate in a live pipe. To be able to function properly, any pipe about to be inspected would have to be drained before AIOM could be deployed in it. Future work on the robot would deal with creating a waterproof design to allow AIOM to function properly in live pipes.

WORKS CITED

[1] N. S. Roslin, A. Anuar, M. F. Jalal. 2012. '*A Review: Hybrid Locomotion of In-pipe Inspection Robot'.* Centre for Advanced Mechatronics and Robotics, Universiti Nasional, Malaysia.

[2] A. Nayak, S. K. Pradhan. 2014. '*Design of a New In-Pipe Inspection Robot'.* NITTTR, Bhopal, India.

[3] H.R. Choi, S. Roh. 2013. '*In-pipe Robot with Active Steering Capability for Moving Inside of Pipelines*'. School of Mechanical Engineering, Sungkyunkwan University, Korea.

[4] A. Reiss. 2012. '*Pipe Robots for Internal Inspection, Non-Destructive Testing and Machining of Pipelines'.* Inspector Systems, Rodermark, Germany.

[5] M.S. Safizadeh, T. Azizzadeh. 2012. '*Automated Detection of Inner Surface Defects in Pipes Using Image Processing Algorithms'.* University of Science and Technology, Iran.

[6] H. R. Halfawy, J. Hengmeechai. 2013. '*Efficient Algorithm for Crack Detection in Sewer Images from Closed-Circuit Television Inspections'.* Universidad Politenica de Valencia.

[7] Y. Rzhanov. 2011. '*Photo-mosaicing of images of pipe inner surface'.* Springer-Verlag, London.

[8] Y. Zhang, G. Yan. 2006. '*In-pipe inspection robot with active pipe-diameter adaptability and automatic tractive force adjusting'.* Shanghai Jiaotong University, Shanghai, China.

[9] S. G. Roh, S. M. Ryew. 2001. *'Actively Steerable Inpipe Inspection Robots for Underground Urban Gas Pipelines'.* Seoul City Gas R&D Centre, Seoul, Korea.

[10] Shivprakash Iyer, Sunil K. Sinha. 2005. *'A robust approach for automatic detection and segmentation of cracks in underground pipeline images'.* Pipeline infrastructure Research Centre, Pennsylvania State University, USA.

[11] Z. Song, H. Ren, J. Zhang. 2016. *'Kinematic Analyses and Motion Control of Wheeled Mobile Robots in Cylindrical Workspaces'.* IEEE Transactions on Automation Science and Engineering.

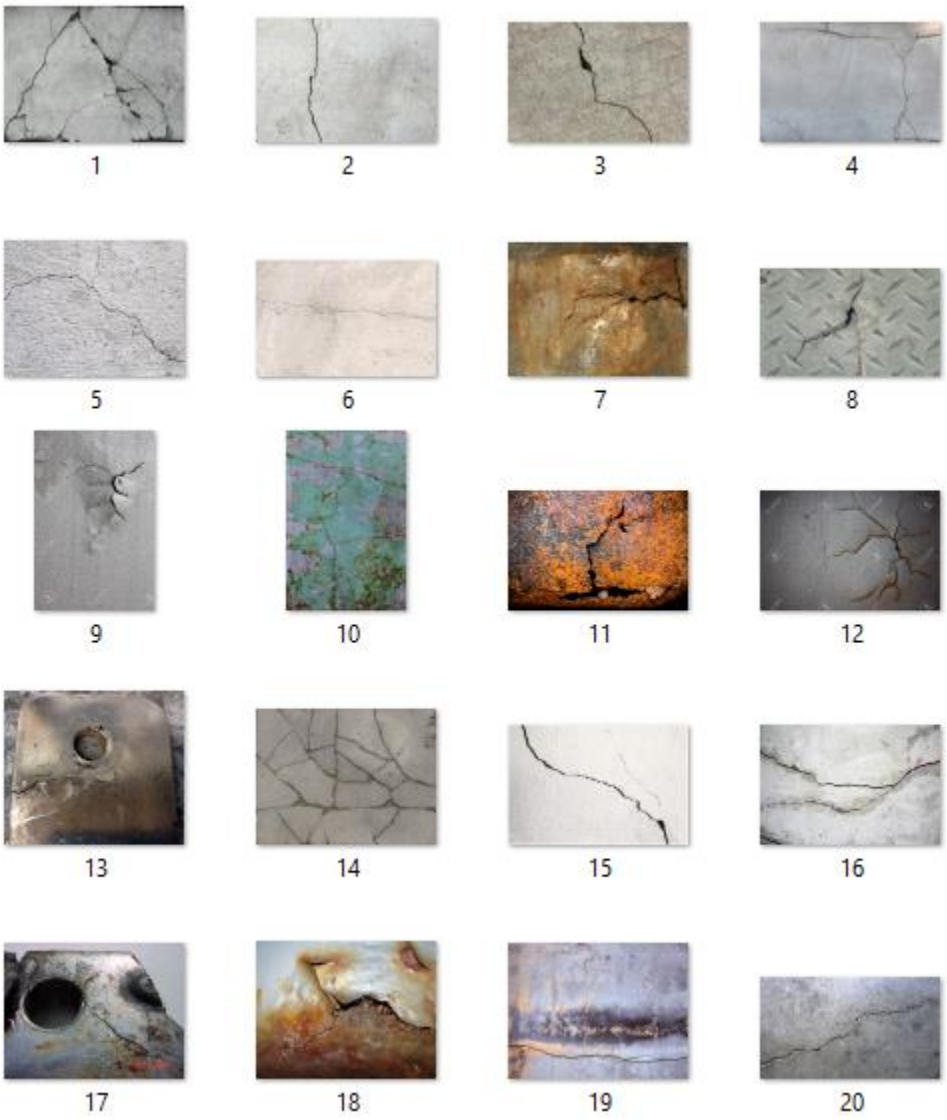
[12] Y. Zhao, Y. Hua, J. Di. 2016. *'Kinematic Modelling and Simulation Analysis of Wheeled Mobile Robot in Round Pipe'.* Dept. of Mech. Engineering, North China University of Technology, Beijing, China.

[13] Parallax Inc. 2011. *'Parallax Continuous Rotation Servo'.* California, USA. Available online at: https://www.parallax.com/sites/default/files/downloads/900-00008-Continuous-Rotation-Servo-Documentation-v2.2.pdf

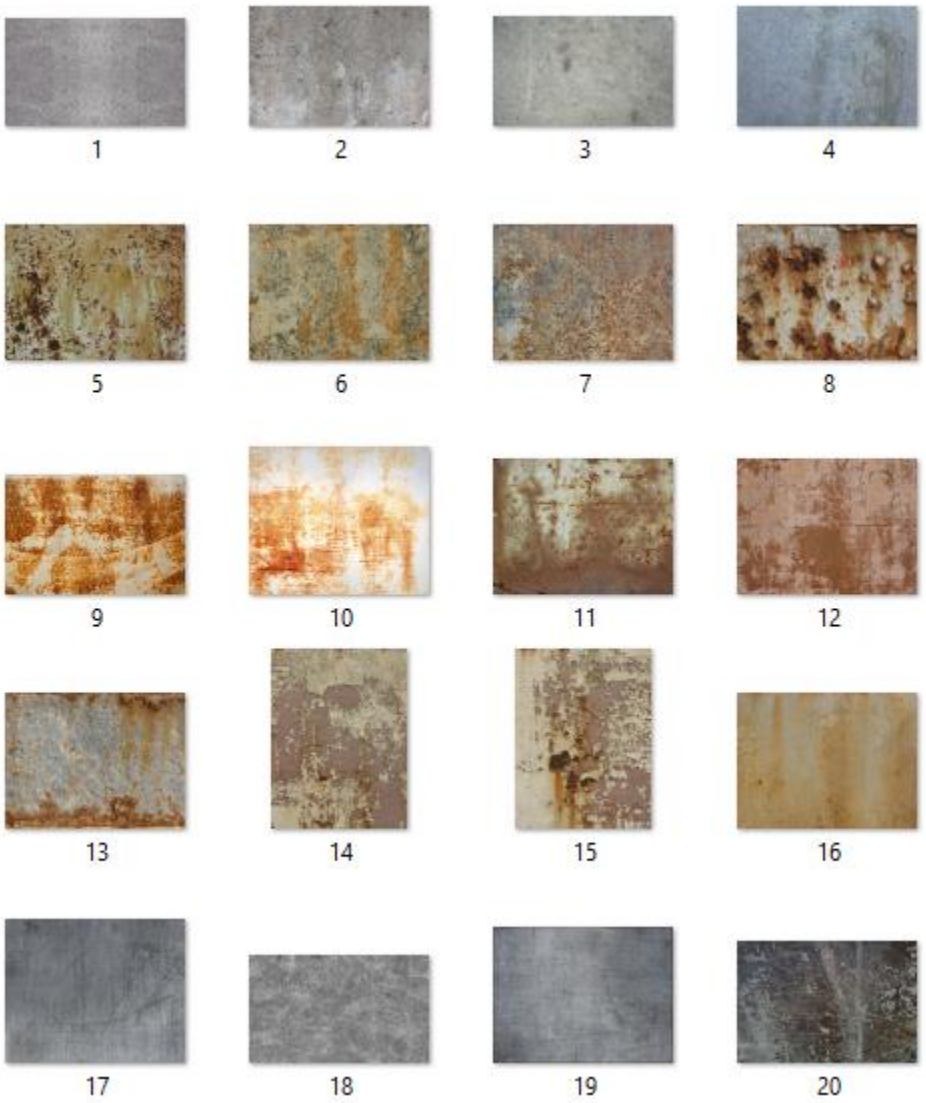
[14] Dotmar Universal Plastics. 2008. '*Coefficient of friction'*. United Kingdom. Available Online at: http://www.dotmar.co.nz/co-efficient-of-friction.html

APPENDIX

APPENDIX I: Crack Image library



APPENDIX II: Non cracked image library



APPENDIX III: Python code for image processing

```
import cv2
import numpy as np
import math


sdfactor=10
meanfactor=2.6
def thresholds(bwimage):
    mean=int(np.mean(bwimage))
    sd=int(np.std(bwimage))
    lowth=sd*sdfactor
    highth=mean*meanfactor
    return lowth, highth


def gettheta(points,center):
    angles=[]
    for p in points:
        angles.append((int(math.atan2((p[1]-center[1]),(p[0]-
center[0]))*57.3)))
    return angles


def distance(point1,point2):
    return math.sqrt((point1[0]-point2[0])**2 + (point1[1]-
point2[1])**2)


def cracksinregion(radius, allcracks, Xcentre, Ycentre):
    i=0
    while i < len(allcracks):
        j=0
        while j<len(allcracks[i]):
            rsquare=((allcracks[i][j][0][0]-
Xcentre)**2)+((allcracks[i][j][0][1]-Ycentre)**2)
            if rsquare<(0.5*radius)**2:
                allcracks[i]=np.delete(allcracks[i],j,0)
```

```
            else:
                j=j+1
        if j==0:
            allcracks=np.delete(allcracks,i,0)
        else:
            i=i+1
    return allcracks

def snip(image, cracks, indices):
    x_max=0
    y_max=0
    x_min=10000
    y_min=10000
    for i in indices:
        temp_x_max=max([a[0][0] for a in cracks[i]])
        if x_max<temp_x_max:
            x_max=temp_x_max
        temp_x_min=min([a[0][0] for a in cracks[i]])
        if x_min>temp_x_min:
            x_min=temp_x_min
        temp_y_max=max([a[0][1] for a in cracks[i]])
        if y_max<temp_y_max:
            y_max=temp_y_max
        temp_y_min=min([a[0][1] for a in cracks[i]])
        if y_min>temp_x_min:
            y_min=temp_y_min
    snipet=image[y_min:y_max, x_min:x_max]
    return snipet

for pic in range(1,2):

    img = cv2.imread('C:\\Users\\Furqan\\Desktop\\Capture2.png')
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    low,high=thresholds(gray)
```

```
    edges = cv2.Canny(gray,low,high,apertureSize = 3)
    cracks=[]
    centre=len(img[0])/2, len(img)/2
    Radius=len(img)/2

contours,hierarchy=cv2.findContours(edges,cv2.RETR_TREE,cv2.CHAIN_APPRO
X_SIMPLE)
    for i in range(0,len(contours)):
        listx=[]
        listy=[]
        for j in range(0,len(contours[i])):
            listx.append(contours[i][j][0][0])
            listy.append(contours[i][j][0][1])
        out=np.polyfit(listx,listy,1,full=True)
        if not out[1]:
            ratio=0
        else:
            ratio=math.sqrt(out[1])/(len(contours[i])**1.4)
        if ratio<0.2 and ratio>0.05 and len(contours[i])>15:
            cracks.append(contours[i])
    cracksofinterest=cracksinregion(Radius,cracks,centre[0],centre[1])
    print len(cracksofinterest)


    mids=[]
    lengths=[]
    for i in range(0,len(cracksofinterest)):
        x=0
        y=0
        l=len(cracksofinterest[i])
        lengths.append(l)
        for j in range(0,l):
            x=cracksofinterest[i][j][0][0]+x
            y=cracksofinterest[i][j][0][1]+y
        mids.append((x/l, y/l))
    i=0
```

```
    sets=[]
    setstemp=[0]
    cracknum=0
    print mids
    while i <len(mids)-1:
        if cracknum not in setstemp:
            setstemp.append(cracknum)
        if distance(mids[i],mids[i+1])<12:

mids[i]=((mids[i][0]*lengths[i]+mids[i+1][0]*lengths[i+1])/(lengths[i]+
lengths[i+1])

,(mids[i][1]*lengths[i]+mids[i+1][1]*lengths[i+1])/(lengths[i]+lengths[
i+1]))
            mids=np.delete(mids,i+1,0)
            lengths=np.delete(lengths,i+1,0)
            cracknum=cracknum+1
            setstemp.append(cracknum)
        else:
            i=i+1
            sets.append(setstemp)
            setstemp=[]
            cracknum=cracknum+1
            setstemp.append(cracknum)
    sets.append(setstemp)


    print mids
    print gettheta(mids,centre)
    print sets
    print max([a[0][0] for a in cracksofinterest[2]])
    #cv2.circle(img, centre, Radius, (200,200,200))
    cv2.circle(img, centre, int(Radius*0.5), (200,200,200))
    #img = img[0:550, 0:727]
    snippet=snip(img, cracksofinterest, sets[0])
    cv2.imwrite('C:\\Users\\Furqan\\Desktop\\snip.jpg', snippet)
    cv2.drawContours(img, cracksofinterest, -1, (0,255,0), 1)
```

```
#for i in range(0,len(mids)):
    #cv2.circle(img ,tuple(mids[i]),3,(0,0,255))
cv2.imshow('rect', img)
cv2.waitKey(0)
```

APPENDIX IV: Pyhton Code For Post-Processing

```
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
import math
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
cracks=[(33,177),(34,179),(34.5,180),(35,180),(35.5,178),(36,177),(36.5
,179),(37,180),(37.5,181),(38.5,180),(39,182),(40,179)]


R=14
L=54
for crack in cracks:

ax.scatter(R*math.cos(crack[1]/57.3),crack[0],(R+2)*math.sin(crack[1]/5
7.3),c='#ff0000')
# Pipe
x=np.linspace(-1*R, R, 100)
y=np.linspace(0, L, 100)
Xc, Yc=np.meshgrid(x, y)
Zc = np.sqrt(R**2-Xc**2)


# Draw parameters
rstride = 4
cstride = 4
ax.plot_surface(Xc, Yc, Zc, alpha=0.3, rstride=rstride,
cstride=cstride)
ax.plot_surface(Xc, Yc, -Zc, alpha=0.3, rstride=rstride,
cstride=cstride)


ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
ax.auto_scale_xyz([-L/2, L/2], [0, L], [-L/2, L/2])
plt.show()
```