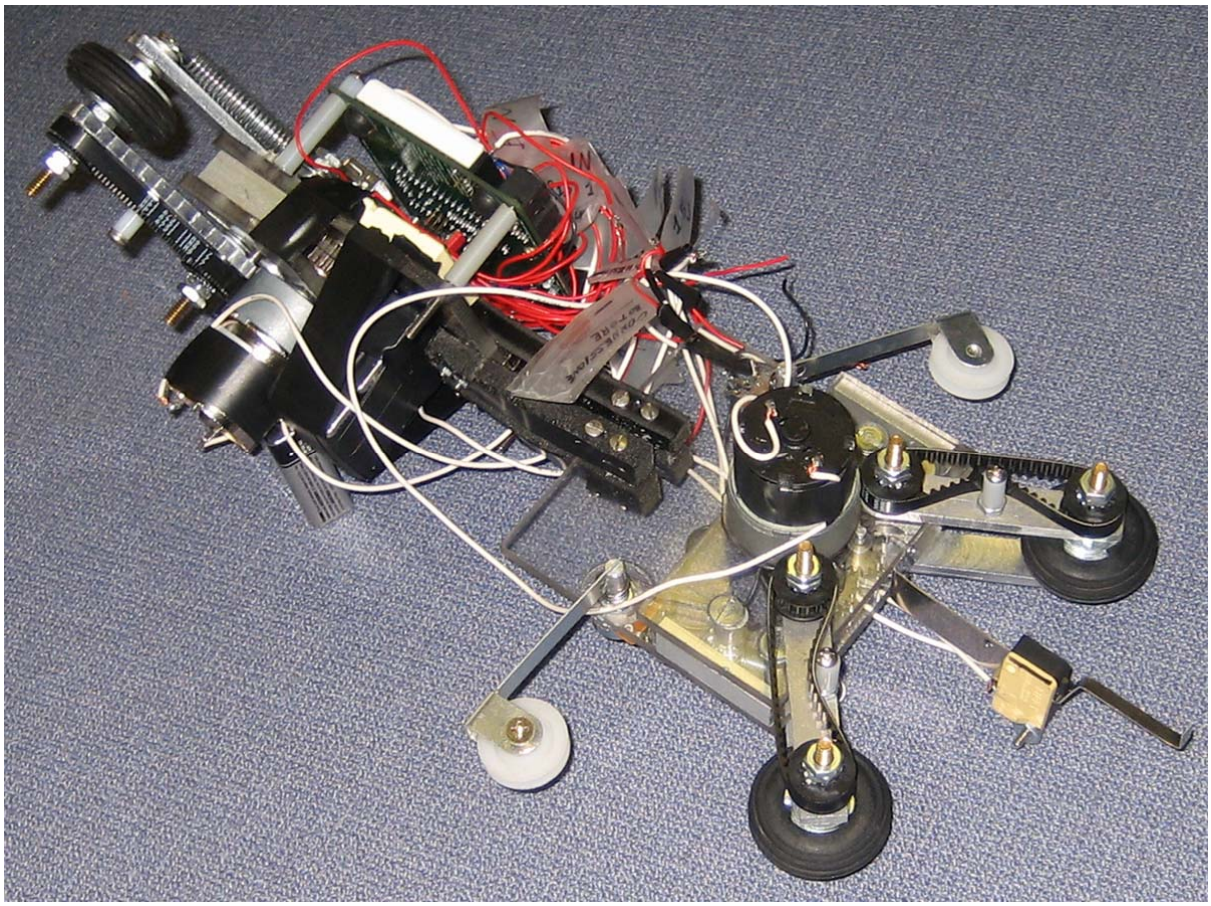
Project report
Mechatronics, Professor V. Kapila
Polytechnic University

Giovanni Del Bufalo, Gianluca Notaro, Alessandro Rovardi

19th May 2008



Abstract

During the Mechatronics Class course at Polytechnic University, New York a group of three people built a robot capable of moving inside pipes and detecting limestone clogs. This paper describes in detail how Cristonia, the robot, was ideated, designed, built and programmed.

Contents

1. Background

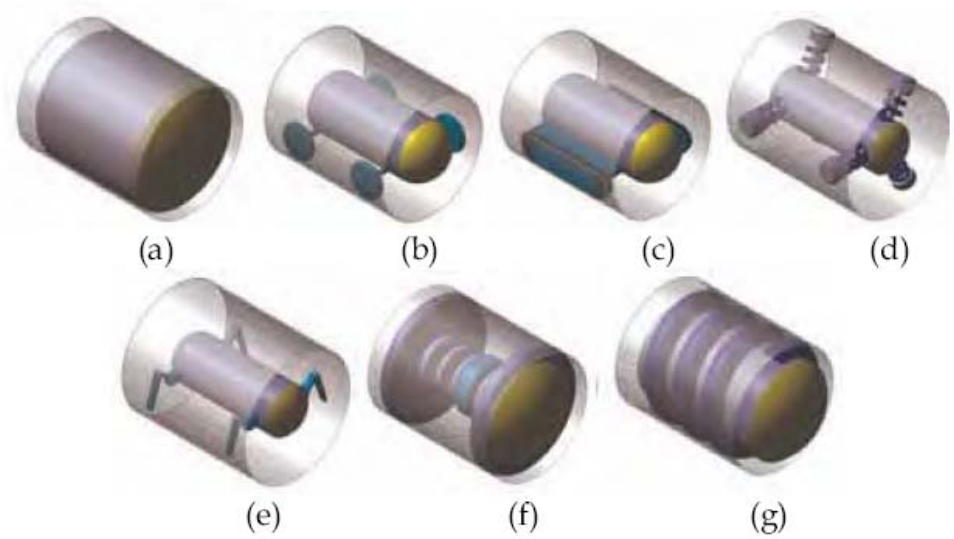
An important part of the *Mechatronics* course at *Polytechnic University* was designing and realizing a robot.

It had to been controlled by a Basic Stamp (BS2) and include sensors and actuators. It should be easily monitored and controlled by humans and safe.

We decided to build an in-pipe inspection robot, and we called it Gennaro.

1.2 Introduction

One of the main concern affecting all buildings and industries is the maintenance required. This implies wasting of time and money necessary to prevent from future damages and to fix those already happened. In particular great problems are associated with the maintenance of pipes conducts. Several approaches have been made in past works to build a vehicle for in-pipe inspection, as shown in the picture below: (a) Pig type. (b) Wheel type. (c) Caterpillar type. (d) Wall-press. (e) Walking type. (f) Inchworm type. (g) Screw type



Numerous authors have studied the problem and several numbers of commercialized robot have been reported up to now for several applications, from gas industry to sewer pipes (Okada & Kanade, 1987; Hirose et al., 1999; Kolesnik, 2002; Suzumori et al., 1998; Kawaguchi et al., 1995; Suzumori, et al., 1999; Tsubouchi et al.,2000; Scholl et al., 2000; Mhramatsu et al., 2000; Ong et al., 2001; Choi & Ryew, 2002; Ryew etal., 2000; Roh et al., 2001; Roh et al., 2002; Roh et al., 2005; Schempf & Vradis, 2005; Schempf,2002; Gamble & Wiesman, 1996).

With this general problem in mind and background of previous studies, we focused our case to hot water pipes. One of the main issues regarding all the tubes that carries hot water is the limestone forming inside them. This can cause not only occlusions in the pipes but also loss of heat exchange and of pressure. For these reasons activities for their maintenance are required. To reduce the high budget needed for it, associated with the detection of the occlusions, the demolition of the pipes, and their substitution we decided to design a versatile robot for the inspection of pipe and detection of limestone inside. One of the future development will possibly be the capability for the robot of repairing the damages directly from inside, saving huge amount of time and money.

1.2 Constraints on the robot

Design of the inspection robot depends on two main critical factors: size and shape of the pipeline. It will weigh strongly on the manoeuvrability of robot and its dimensions. An ideal robot should:

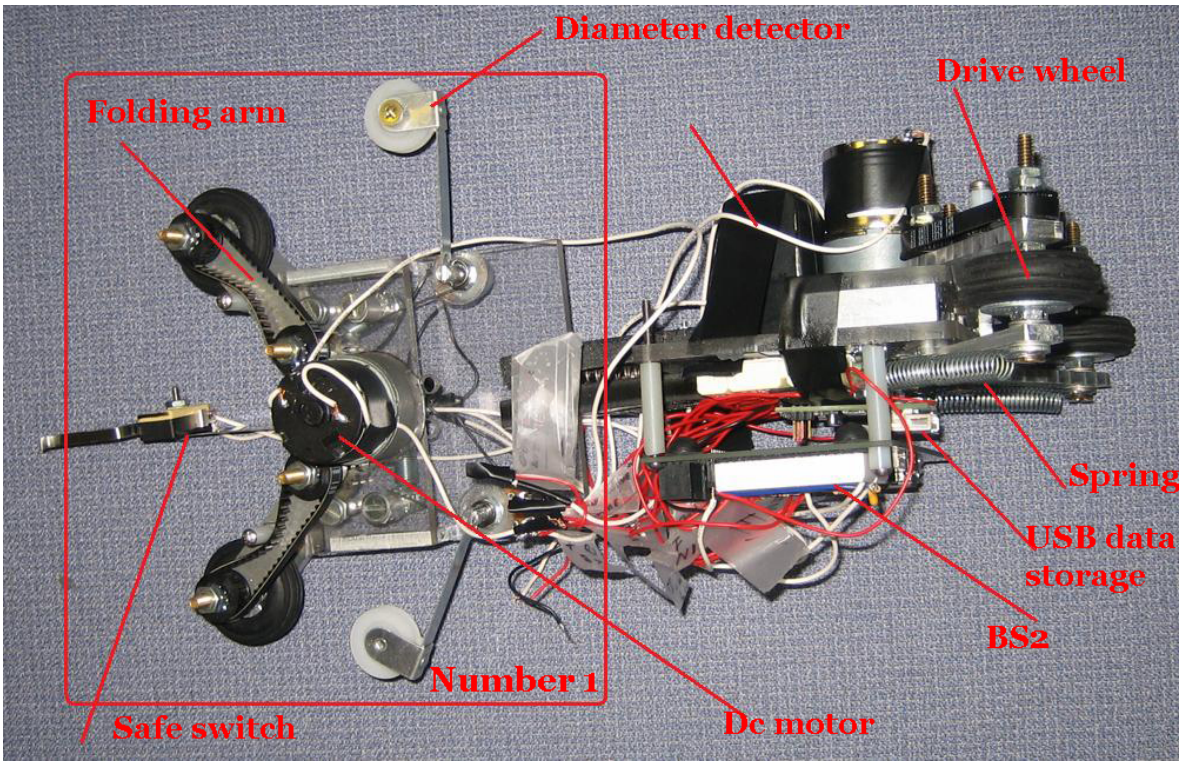
1. drive through a pipe that can change its diameter along his pattern;

2. cope with elbows and branches, reducer, valves with unexpected mechanical damages that could change its mechanical configuration;

3. have sufficient traction to move and to carry out tasks as measurements or clogging detection in a slippery and not plane surface as a pipe

4. be robust and reliable

2. Mechanical design



Several constrains were taking into account in the first phase of the mechanical design. They were:

- minimal and maximal dimensions
- weight
- moving ability
- power request
- cost issues

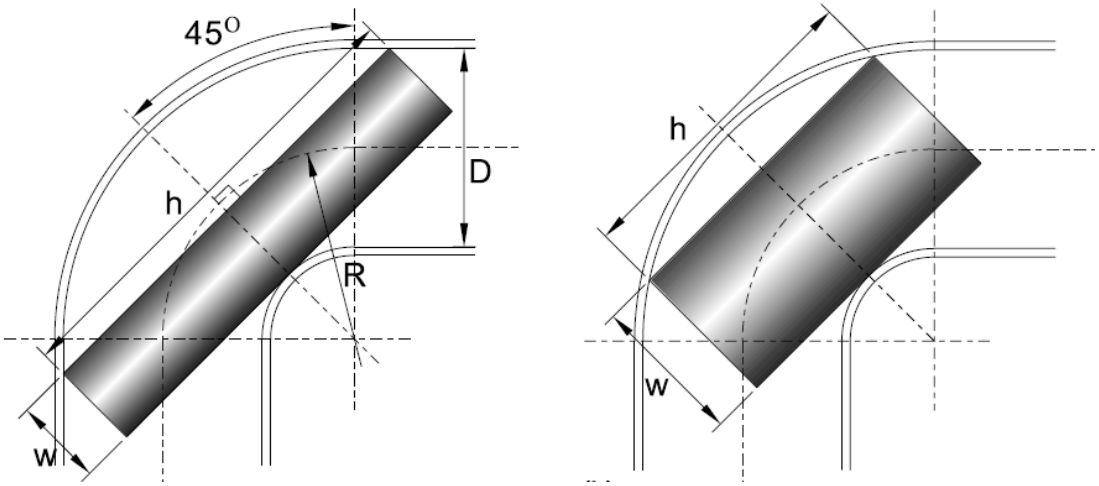
Any of the above influenced the others and was sometimes in contradiction. We wanted a light robot, with high power and torque to move easily and firmly, and it must work in the small space given by the 6 inch tube pipe. These were the main features for the robot we had in mind, but being realistic we knew from the beginning that we will have to arrive to a compromise between them to be successful.

2.1minimal and maximal dimensions

The maximal dimension was given by the nominal pipe diameter we wanted to expect, of six inch. In addition, the robot was build to inspect pipes clogged by limestone, so we decided that it should be able to move in pipes of up to 5 inch of inner diameter. One

fundamental and critical aspect were corners. As shown in the picture below, the width (w) and high (h) of the robot influences each other, and the following formula[1] has been used to design the robot:

$$0 < w \leqslant \{(R + D/2)\sin 45° - (R - D/2)\},$$



The minimal acceptable dimension instead was given by the room necessary to equip the robot with a Basic Stamp, all the circuitery, the data acquisition devices, the sensors and motors and the power supplies needed.

2.2 Weight

Weight was another critical parameter. A light robot was desired in order to need less power to move, to be agile and to run in vertical pipes. However, this factor was influenced by the motor's choice and the batteries needed. Approximately 50% of the total weight was caused by the batteries and motor. However any other kind of propulsion or power supply we thought was discarded, and this seems to be the only possible way to have a compact and self propelled vehicle.

2.3 Moving ability

The moving ability was probably the greatest problem to deal with. The smooth surface of the pipes was something difficult to cope with and the round surface adds further difficulties. The robot was intended to move not only in horizontal but also in vertical pipes, so it has to hold onto the surfaces and have the necessary grip and power to climb them. All this requirements had to face with the reality of the chosen traction engine, the DC motors, and with our limited budget. We decided to use four rubber tyres of 1 inch of diameter, because they showed good grip, were light, cheap and compatible with the robot dimension. As will be better explained later we decided to have a robot made up of two

[1] H.R. Choi, S.M. Ryew. *Robotic system with active steering capability for internal inspection of urban gas pipelines.* Mechatronics 12 (2002) 713–736

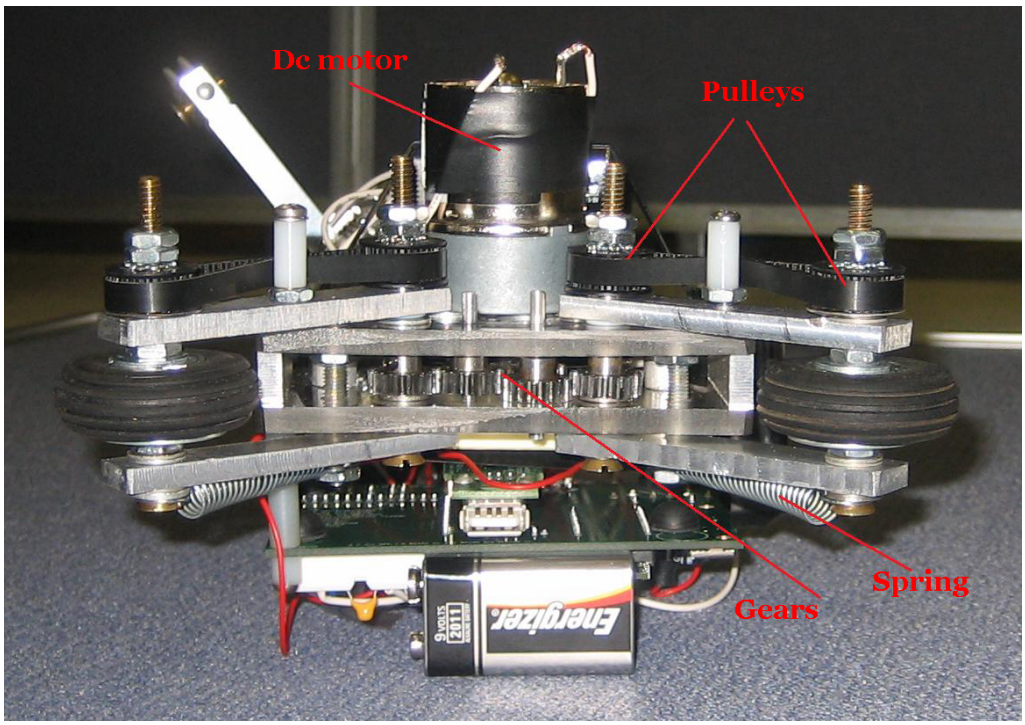
different autonomous parts. In order to link them, four strips of rubber with approximate dimensions of *5 cm* length *1 cm* width and *1 cm* height rubber were made, and fixed to the frame. The rubber was reinforced with a linear spring to obtain the desired flexibility and rigidity when approaching an elbow.

2.4 Power request

The motors used work with a 12 V dc current. They have a power request of approximately 1W. The Basic Stamp will be independent and equipped with a 9V battery. We decided to use 9V batteries in parallel also for the motors, in order to provide the sufficient current requested, but this added lot of weight to the robot. One future improvement will be to use more efficient power supply with rechargeable lithium batteries. However, the last period is good and the robot will be able to work for about 30 minutes before of batteries discharge.

2.5 Frame and transmission

The body of robot is designed to be compact and flexible and to have enough traction to climb vertical pipes. Mechanically the robot is made up of two autonomous and identical driving vehicle, connected by a flexible material. We will call them "*number 1*" and "*number 2*" . We chose to build a robot with only two vehicle, but this has been done only for budget and time issues. It is possible to link as many vehicle as you wish, and this is preferable if more room is needed. Indeed, the more vehicle are linked the best will be the dynamical behaviour of the robot.

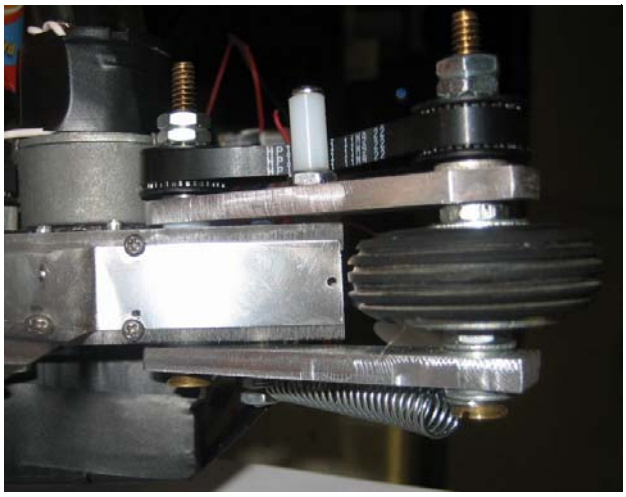


The robot has only four wheels in total that are in contact with the inner wall of the pipe, two in the first robot and two in the second. Each robot frame is made of a Plexiglass

square sheet of a quarter of inch of thickness that houses the dc motor, and to which are fixed the inner edge of two folding arms. On the other edges of the arms, made in aluminium, are mounted the wheels. The two frame are rotated of 90 degrees in order to have a wheel every 90 degree and we chose the Plexiglas to reduce weight and for its low price and easy machinery. For the same reasons we have chosen the aluminium when better mechanical performance were required.

The mechanical transmission drives the vehicle thanks to gears and flexible belts. Since the two wheels of each independent part have to turn in opposite directions (one clockwise, the other contraclockwise) to move the robot, we used four aluminium gears from the pinion gear of the dc motor. Dimensions of the aluminium gear are the same and therefore the gear ratio and torque is the same. The externals gears are mounted on a shaft that pass through the inner edge of the moving arms and the Plaxiglass frame, provided of ball bearings. The arms are free to move around their shaft and a spring is used to let them open and tightly fixed to the internal surface of the pipe. A pulley is fixed to the same axle, and through a rubber belt with trapezoidal teeth it moves another pulley at the other end of the arm and the wheel, fixed with this second pulley.

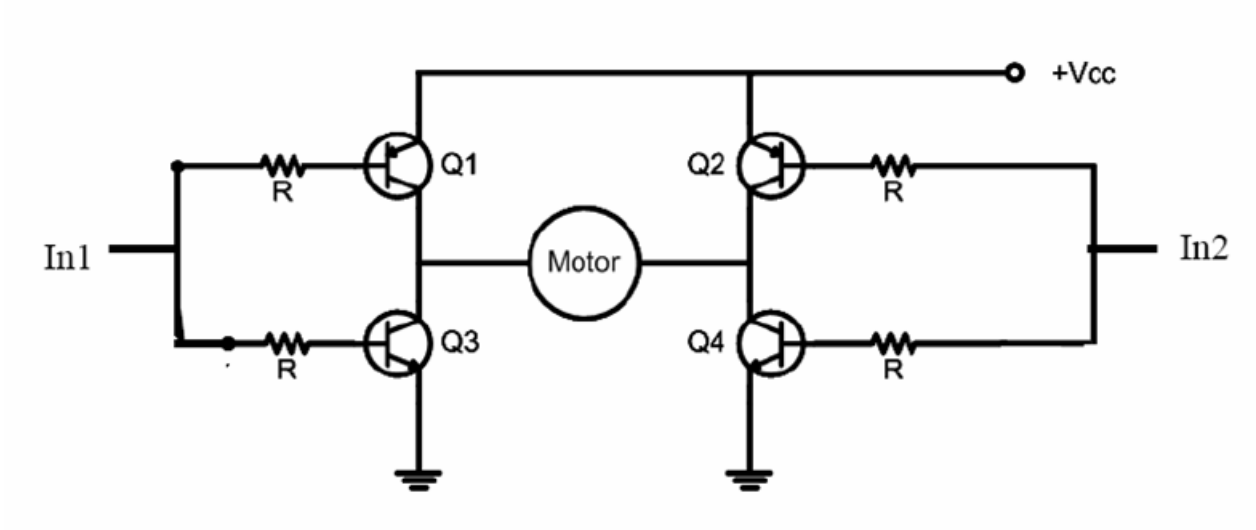
We decided to use this kind of transmission in order to have two wheels driven by the same motor and mounted on folding arms that can cope with different diameter of the pipes. We used bearings and grease to reduce the unavoidable mechanical loss.



2.6 Motors

The robot is equipped with two motors, one on *number 1* and one on *number 2*. Two identical motors were ordered from Jameco and since we didn't need high speed but high torque we chose two 12 V dc Reversible Gear Head Motors. They work at 12V and at approximately *60 rpm*. Unfortunately one of the motors didn't work well and we didn't have time to replace with an identical one. We replaced it with a similar but not same, that worked at different *rpm*. They exhibited good torque but needed lot of current and energy supply. To activate them, we will use two full bridge and pulse-width-modulation (PWM),

as shown in the scheme below. Each bridge has two inputs to control the base leads of the pair of transistors.



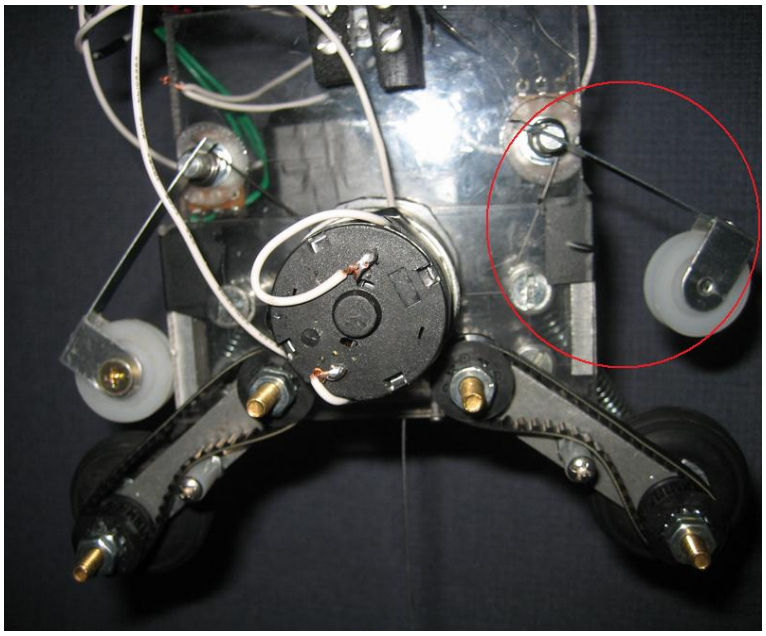
Unfortunately, due to the changing of the motor, the H-bridge chosen didn't work properly. To let them work at the same speed ratio then, and to have the same angular velocity for the wheels, we tried to use a potentiometer to adjust *number 2* at lower voltage, at approximately 9 V. This is only a partial solution that will be fixed in the future.

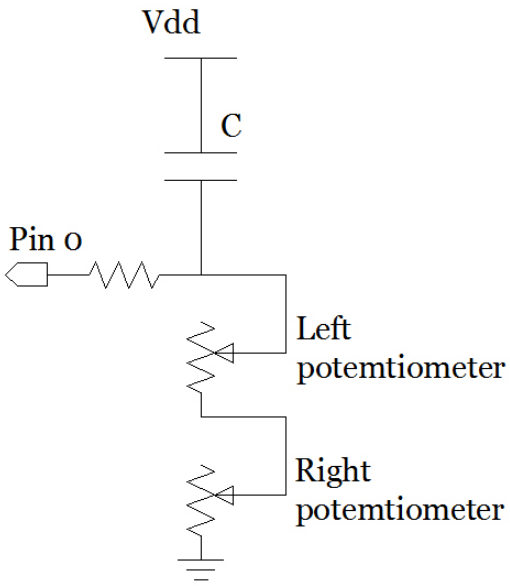
3. Sensors and robotics

The robot is provided with different sensors and robotics:

- two potentiometers
- one data acquisition device
- one relay
- two H-bridge to control the motors

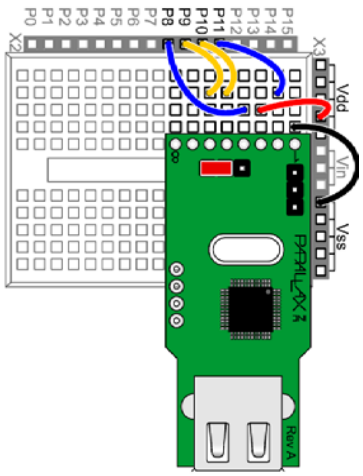
We applied to *number 1* two angular potentiometers connected to two additional arms. They are pushed outward by two springs to stay always attached to the internal surface of the pipe. Using a RCTIME command on each arm, we are able to correlate the movement of the sensors to the measure of the diameter and therefore we can measure the inner diameter of the pipe.

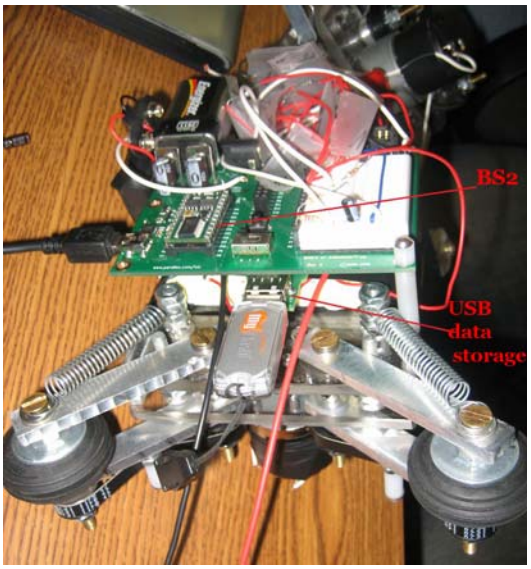




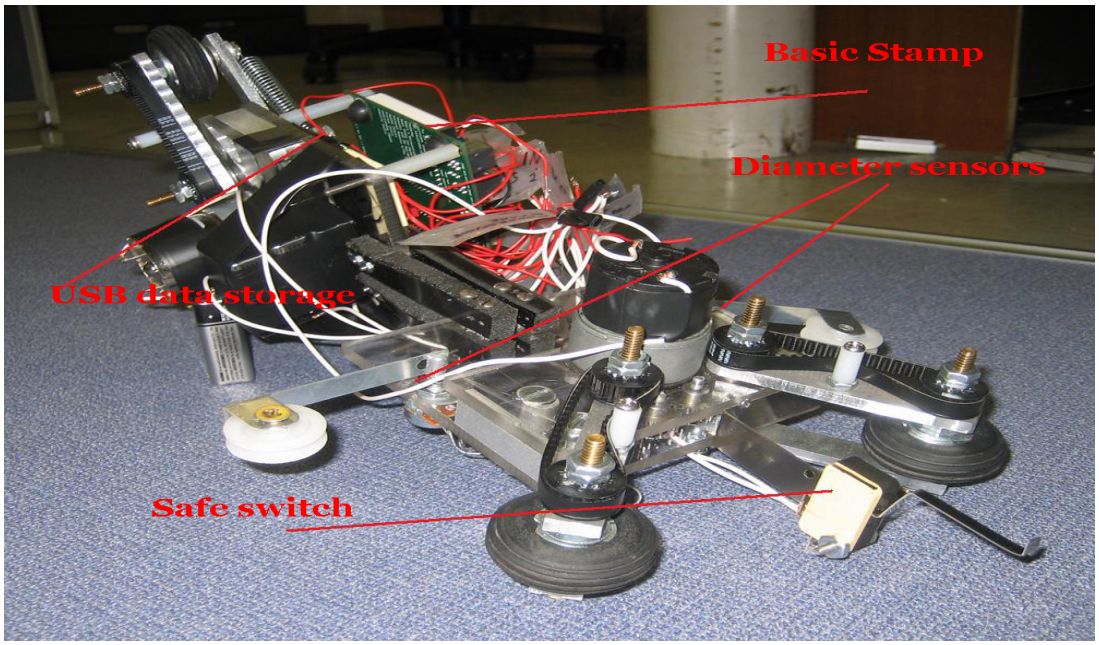
A common problem for inspection pipe robots is the communication with the external operator. One possible way is to use tether cables that unfortunately is not versatile for long and articulated pipelines. Another possible way is to use wireless signal, but it has the main problem in the reliability of the signal, especially with metal pipes. We decided to provide our robot with a third solution, a data logger system that can store data and make these available to the external operator when it goes out the pipelines. It is composed of a Memory Stick Datalogger (#27937). The Memory Stick Datalogger is a USB Host Bridge which allows you to connect a USB Mass Storage Device, such as a Thumb Drive, to your BASIC Stamp®, SX or Propeller Microcontroller. The Vinculum IC/Firmware on the

Datalogger Module handles the File System of the Memory Stick so that you can share the files with your PC. All of this control using simple Serial commands. Stored data can be downloaded and elaborated with common software like Matlab or Excel to take decisions about the next maintenance period.





The switch is mounted to the top of the robot and is therefore pressed when it finds an obstacle on its way. It is intended as a safety equipment: if the robot fall into an obstacle such as a blocked pipe, the switch is pressed. The robot is programmed to stop, go reverse and try to go strait again. It repeats this procedure three times, after that it proceeds backward to the operator.



4. Cost analysis

The total cost of the robot was an important factor. We tried to reduce costs but the number of components and some parts as motors or data logger were absolutely necessary. However if a set of more than two vehicle is build, the unit price per vehicle will be lower, and will be possible to contain costs. Below are reported the description of the parts used, where do we bought them, their unit price and the total amount.

Description	*Company*	*Quntity*	*Price per Unit $*	*Total Price $*
Basic stamp	Parallax	1	59	59
Data logger usb	parallax	1	35	35
DC Motor Gear ReliaPro	Jameco	2	20	40
H-bridge	Jameco	2	3.5	7
Pinion gears	HPI Racing	10	3	30
Transmission Belt	HPI Racing	4	5	20
Plastic Pulley	HPI Racing	8	3	24
Linear Spring	Sid's Hardware	4	0.25	1
Rubber Wheels	HPI Racing	4	2	8
Bearing	HPI Racing	8	4	32
Potentiometer	Jameco	2	0.5	1
Mechanical Relay	Jameco	1	0.5	0.5
Plexiglass Sheet	Sid's Hardware	1	3	3
Screw	Sid's Hardware	30	0.3	9
Aluminum Sheet	Sid's Hardware	1	5	5
Linear Spring	Sid's Hardware	4	0.25	1
Batteries	Energize	10	2	20
Total		94		296.5

5. Construction, locomotion and future improvements

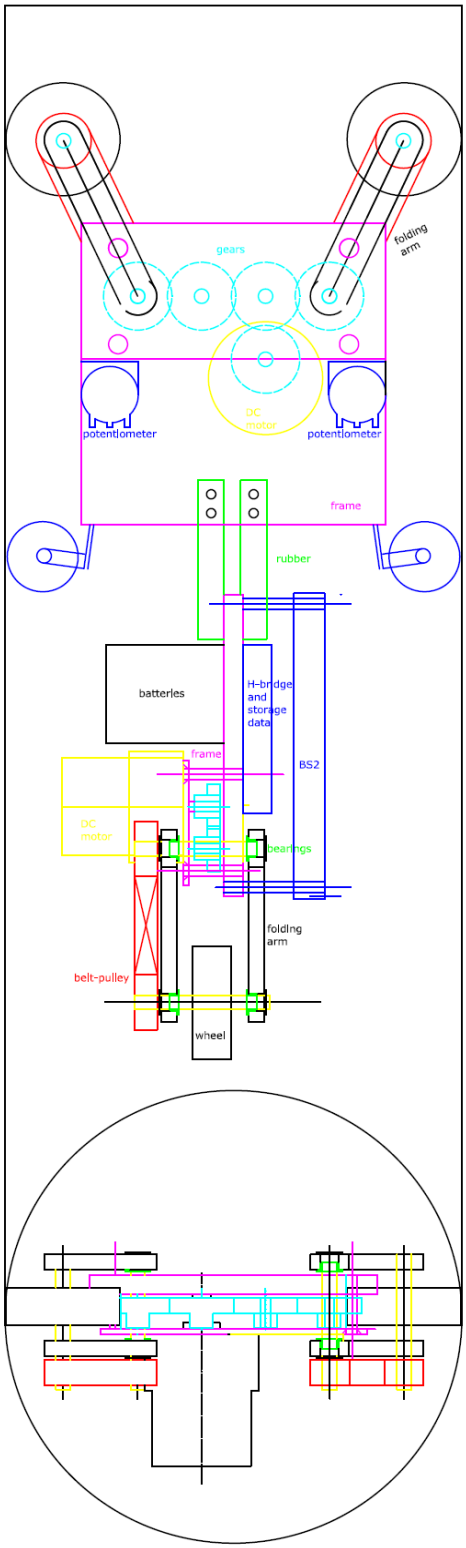
During the construction and assembly of the robot we found some critical parts that need to be explained in more detail. The first is about the folding arms. At first we decided to have only one cantilever beam per wheel, using a square aluminum sheet (dimensions 60mm x 2mm x 0.250 mm). The main problem was how to lock it firmly to the frame, since it had to sustain all the weight of the robot, and leave at the same time the freedom to bend. In addition it had to leave the shaft with the pulley free to spin inside it. As soon as we finished it we realized that it was not stiff enough to deal with the transmission and keep the robot tight to the pipes surface. Then we decided to use two aluminum cantilever for each wheel, one for each side of the wheel, and in this second version the robot behave much better and worked properly. However it probably remains the most critical part for the mechanical system, for their intrinsic complexity and for the small tolerance related with their dimension. Another possible enhancement for a new version in the future will be the spring system for the folding arms. We chose to use one linear spring for each arm because of the simplicity and easiness to assemble it. However the main problem is the bending moment resultant on each arm for the spring, in a complex and delicate component as the folding arms. Different solutions have been considered, as other ways to link them from the frame to the arms, or using torsion springs instead of linear one, but they have not been tried because too hard to been realized in short time.

About the locomotion, the power delivered by the motors seems enough for our needs, however as told before a better power supply is required for an enhanced version. The main problems are narrow elbows, because sometimes the robot collide the wall and blocks itself. This is due mainly to its shape full of corners. A necessary enhancement will be to build a smooth body to house the frame and the electrical components. This seems something necessary also considering the environment in which it will operate. The contact with liquids (water) and debris present in the pipes is something to avoid for the circuitery and the gear system, and a waterproof shell seems indispensable for the robot.

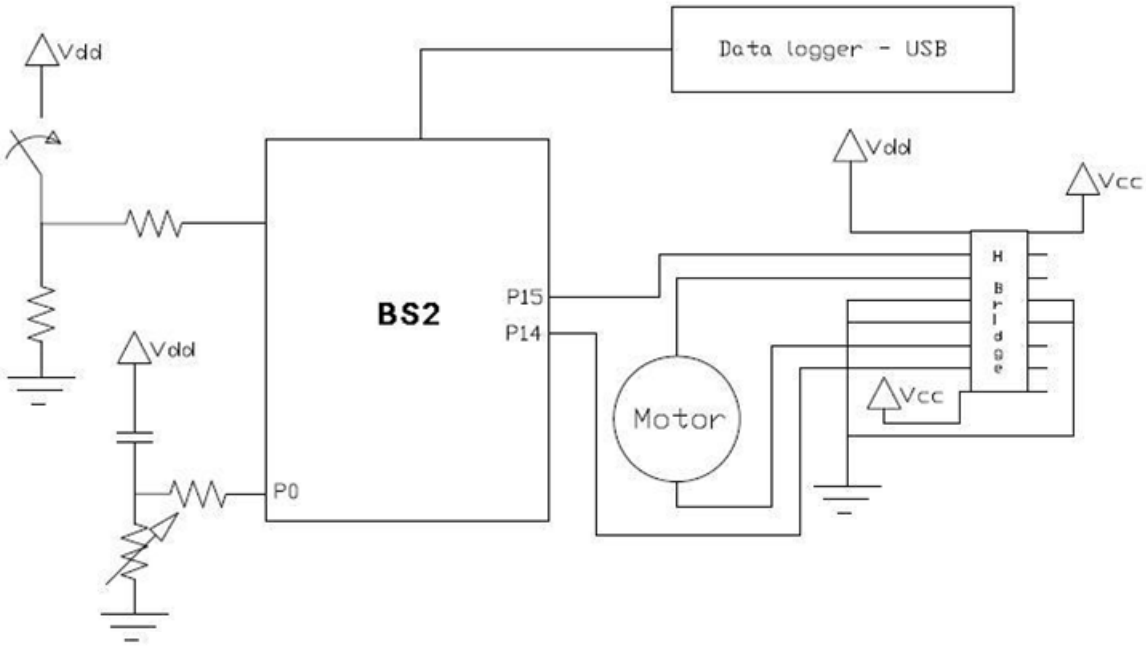
Several other improvements are possible for future works, from cameras and lights to see the interior of the pipes to wireless communication, and grinding machines to erode the limestone, and this project is only the first stone for a bigger figure.

6.Appendix

Autocad design



Circuitery



PBasic code

```
' {$STAMP BS2}

' {$PBASIC 2.5}

i VAR Byte

rele PIN 2

MOTORINPUT1 PIN 15

MOTORINPUT2 PIN 14

TIMEOBSTACLE VAR Byte

COUNTOBSTMAX CON 255

TIMEOBSTACLE=0

' -----[ I/O Definitions ]--------------------------------------------------

Sensor        PIN     0                      ' Light Sensor Or Thermistor

TX            PIN     8                      ' Transmit Data   --> 27937.4 (RXD)

RTS           PIN     9                      ' Request To Send --> 27937.6 (CTS)
```

```
RX          PIN    10                    ' Receive Data    <-- 27937.5 (TXD)
CTS         PIN    11                    ' Clear To Send   <-- 27937.2 (RTS)
' -----[ Constants ]-------------------------------------------------------
Baud        CON    84                    ' Serial Baud Rate 9600 bps (BS2)
NumSamples  CON    10                    ' Number Of Samples To Log
' -----[ Variables ]-------------------------------------------------------
buffer      VAR    Byte(15)              ' Input Buffer
index       VAR    Byte                  ' Index Variable
ioByte      VAR    Byte                  ' Input/Output Storage
counter     VAR    Word                  ' Counter Variable
result      VAR    Word                   ' Sensor RCTIME Result
work        VAR    Word                  ' Work Variable
flag        VAR    Bit                   ' Event Status Flag
' -----[ Initialization ]--------------------------------------------------
counter=0
PAUSE 200                                ' Allow Time To Settle
HIGH TX                                  ' Initialize Transmit Line
LOW RTS                                  ' Take Vinculum Out Of Reset
PAUSE 600                                ' Allow Time To Settle
DO
        SEROUT TX\CTS, Baud, ["E", CR]       ' Sync Command Character
        GOSUB Get_Data                        ' Get Response
        PAUSE 250
LOOP UNTIL ioByte = $0D                  ' Wait For Carriage Return
DO
        SEROUT TX\CTS, Baud, ["e", CR]       ' Sync Command Character
        GOSUB Get_Data                        ' Get Response
```

```
        PAUSE 250
LOOP UNTIL ioByte = $0D                    ' Wait For Carriage Return
SEROUT TX\CTS, Baud, ["SCS", CR]           ' Switch To Short Command Mode
GOSUB Get_Data                             ' Purge Receive Buffer
SEROUT TX\CTS, Baud, [$07, $20, "datafile.txt", CR]
GOSUB Get_Data                             ' Purge Receive Buffer
                                           ' Then Create File
SEROUT TX\CTS, Baud, [$09, $20, "datafile.txt", CR]
GOSUB Get_Data                             ' Purge Receive Buffer

DEBUG "Done", CR, "Switching to Short Command Mode..."
SEROUT TX\CTS, Baud, ["SCS", CR]           ' Switch To Short Command Mode
GOSUB Get_Data                             ' Purge Receive Buffer
DEBUG "Done!", CR, "Waiting for Memory Stick..."
Check_Drive:
 DO
        SEROUT TX\CTS, Baud, [CR]                  ' Prompt Device For Status
        GOSUB Get_Data                             ' Purge Receive Buffer
        IF buffer(0) = ">" THEN                    ' Check For Ready Prompt
        EXIT                                       ' If Ready Then Exit Loop
        ELSEIF buffer(0) = "N" AND buffer(1) = "D" THEN
        DEBUG "."                  ' Device Ready But No Memory Stick
        ELSEIF buffer(0) = "D" AND buffer(1) = "D" AND flag = 0 THEN
         DEBUG "Connected!", CR, "Accessing..."
        flag = 1                                    ' Memory Stick Ready
        ELSE
        DEBUG "."
```

```
        ENDIF
        PAUSE 250                                   ' Command Retry Delay
LOOP
DEBUG "Ready!", CR
DEBUG "Opening Data File..."                        ' First Delete File
SEROUT TX\CTS, Baud, [$07, $20, "datafile.txt", CR]
GOSUB Get_Data                                      ' Purge Receive Buffer
                                                    ' Then Create File
SEROUT TX\CTS, Baud, [$09, $20, "datafile.txt", CR]
GOSUB Get_Data                                      ' Purge Receive Buffer
acc:
PAUSE 5000
FOR i = 50 TO 1
DEBUG "acc"
LOW  MOTORINPUT1
HIGH MOTORINPUT2
PAUSE 10
LOW  MOTORINPUT1
LOW  MOTORINPUT2
PAUSE i
NEXT
main:
IF rele=1 THEN
GOTO obstacle
ELSE
DEBUG "go"
LOW   MOTORINPUT1
```

```
HIGH  MOTORINPUT2
ENDIF
DEBUG "Open!", CR, CR, "Writing Data...", CR
counter = counter + 1                    ' Number Of Samples To Log
HIGH Sensor                              ' Charge Capacitor
PAUSE 1                                  ' Wait 1 ms
RCTIME Sensor, 1, result                 ' Measure Discharge Time
DEBUG "Sample ", DEC5 counter, "  ", DEC5 result, CR      ' Display Results
SEROUT TX\CTS, Baud, [$08, $20, $00, $00, $00, $0D, CR,
DEC5 counter, ",", DEC5 result, CR, LF, CR]
PAUSE 500                                ' Write Results/Delay
GOSUB Get_Data                           ' Purge Receive Buffer
'NEXT
GOTO main
```

obstacle:

```
TIMEOBSTACLE=TIMEOBSTACLE+2
DEBUG "obstacle!"
DEBUG HOME
LOW  MOTORINPUT1
LOW  MOTORINPUT2
PAUSE 2000
FOR i = 50 TO 1
DEBUG "go  reverse"
DEBUG HOME
HIGH MOTORINPUT1
LOW  MOTORINPUT2
```

```
PAUSE 15
LOW  MOTORINPUT1
LOW  MOTORINPUT2
PAUSE i
NEXT
FOR i = 1 TO countobstmax
DEBUG "rwd"
DEBUG ? i
DEBUG HOME
HIGH MOTORINPUT1
LOW  MOTORINPUT2
NEXT
PAUSE 2000
IF timeobstacle < 1 THEN
FOR i = 50 TO 1
DEBUG "go!"
DEBUG HOME
HIGH MOTORINPUT2
LOW  MOTORINPUT1
PAUSE 15
LOW  MOTORINPUT2
LOW  MOTORINPUT1
PAUSE i
NEXT

FOR i= 1 TO countobstmax
IF rele = 0 THEN
```

```
DEBUG "FRW"
DEBUG HOME
HIGH MOTORINPUT2
LOW  MOTORINPUT1
ELSE
GOTO obstacle
ENDIF
NEXT
GOTO main
ELSE
GOTO gohome
ENDIF
gohome:
DEBUG "Closing Data File..."                    ' Close File (MUST CLOSE!)
SEROUT TX\CTS, Baud, [$0A, $20, "datafile.txt", CR]
GOSUB Get_Data                                  ' Purge Receive Buffer

DEBUG "Done!", CR, CR, "Opening Data File..."
SEROUT TX\CTS, Baud, [$0E, $20, "datafile.txt", CR]
GOSUB Get_Data                                  ' Purge Receive Buffer
                                                ' Open Fil For Read Mode
DEBUG "Open!", CR, "Reading Data...", CR
FOR counter = 1 TO NumSamples                   ' Number Of Samples To Read
SEROUT TX\CTS, Baud, [$0B, $20, $00, $00, $00, $0D, CR]
SERIN RX\RTS, Baud, [DEC5 work, ioByte, DEC5 result, ioByte, ioByte]
'GOSUB Get_Data                                 ' Purge Receive Buffer
DEBUG "Sample ", DEC5 counter, " of ", DEC5 NumSamples,
```

```
" --> ", DEC5 result, CR                          ' Display Results
PAUSE 500                                         ' Delay
NEXT
DEBUG "Closing Data File...Program Complete!"
SEROUT TX\CTS, Baud, [$0A, $20, "datafile.txt", CR]
GOSUB Get_Data                                    ' Purge Receive Buffer
DO
DEBUG "HOME"
HIGH MOTORINPUT1
LOW  MOTORINPUT2
LOOP UNTIL rele = 1
STOP
Get_Data:
index = 0                                         ' Reset Index Pointer
DO                                                ' Receive Data
SERIN RX\RTS, Baud, 100, Timeout, [ioByte]
buffer(index) = ioByte                            ' Add Received Byte To Buffer
index = index + 1                                 ' Increment Index Pointer
IF index > 14 THEN Timeout                        ' Check For Overflow
LOOP
Timeout:
 RETURN
```