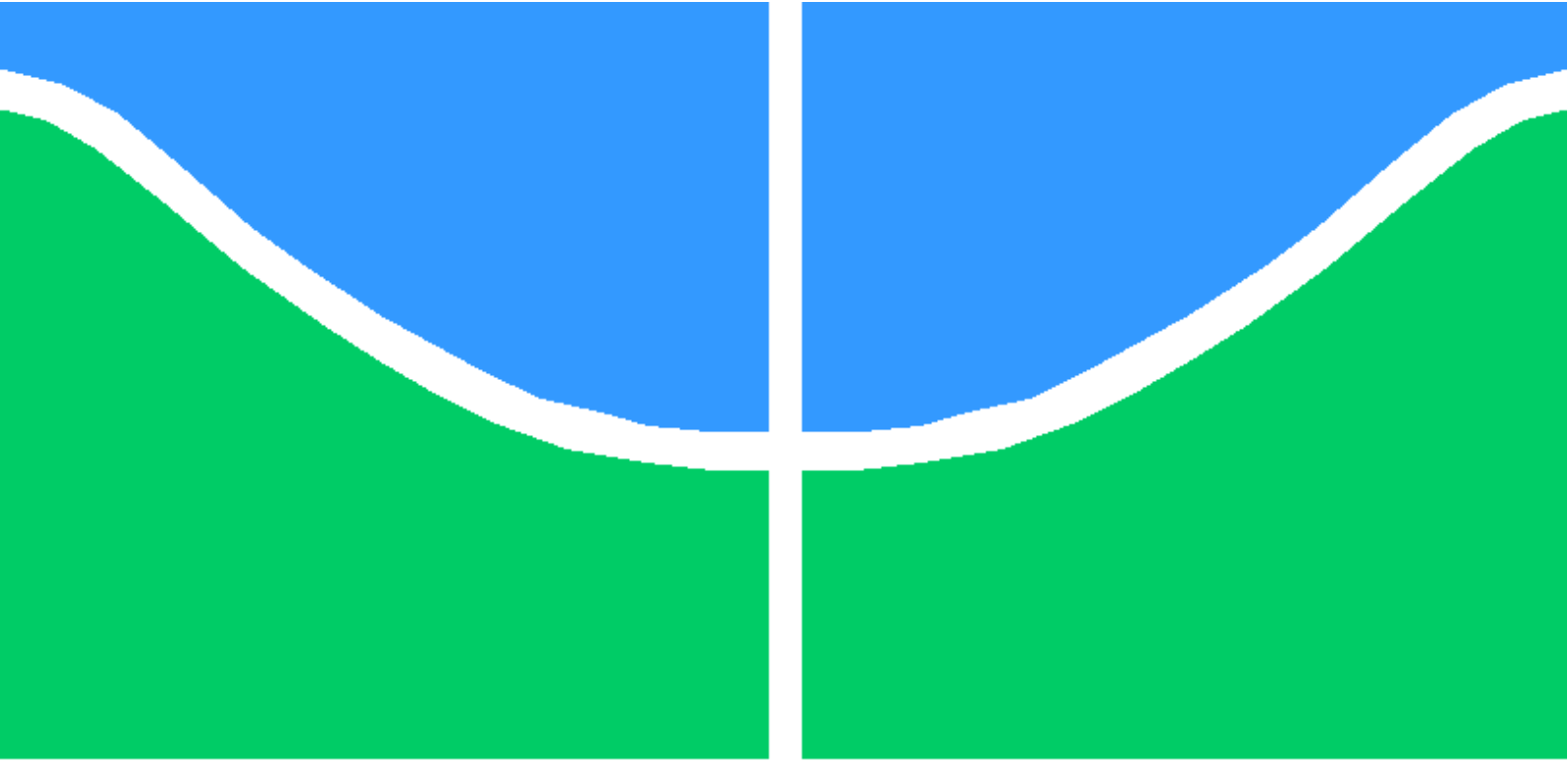


DISSERTATION

Sensors in Modular Robotics for Pipeline Inspection: Design and Test of Erekobot-σ Module

Ana Carolina Cardoso de Sousa

Brasília
November, 2014

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

DISSERTATION

Sensors in Modular Robotics for Pipeline Inspection: Design and Test of Erekobot-σ Module

Ana Carolina Cardoso de Sousa

Report submitted to the Department of Mechanical Engineering as a partial requirement for obtaining the title of Master in Mechatronic Systems

Examination board

Carla Maria Chagas e Cavalcante Koike, CIC/UnB _____
Advisor

Carlos Humberto Llanos Quintero, ENM/UnB _____
Chair member

Renato Alves Borges , ENE/UnB _____
Chair member

To

My family.

Ana Carolina Cardoso de Sousa

Acknowledgments

Here, I am using the opportunity to express my gratitude to everyone who supported me throughout these years. I am thankful for their aspiring guidance, constructive criticism and friendly advices during my project, not only in the academic sense, but also in a personal sense. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to the project.

I express my warm thanks to my academic advisors Dr. Carla Koike and Dr. Dianne Magalhães for their expertise, understanding, and patience in the Ereko Group, they provided me with the required facilities and conductive condictions for my dissertation project. I appreciate their vast knowledge and skill in many areas (e.g., robotics, computer science, electronics, mechanics, interaction with participants and contributors), and their assistance in writing reports (i.e., proceedings articles and this dissertation). I also want to express my gratitude to all participants of the Ereko Group, especially Ricardo, whose volunteer work was fundamental for this project. I would also like to thank the other members of the chair, Dr. Carlos Humberto Llanos Quintero and Dr. Renato Alves Borges for the careful evaluation of my project.

I must also acknowledge the Brazilian institutions: Agência Nacional do Petróleo, Gás Natural e Biocombustíveis (ANP), Financiadora de Estudos e Projetos (FINEP), Ministério da Ciência e Tecnologia (MCT), and Petróleo Brasileiro S.A. (Petrobras) for financial support in the program PRH-PB 223, especially in the present study. I would also like to thank the coordinator of the program Dr. Eugênio Liborio Feitosa Fortaleza for providing me support and knowledge. I also express my eternal gratitude to the University of Brasilia (UnB) and the Department of Mechanical Engineer (ENM); I will always consider these places home.

Finally, I would also like to thank my parents for the support they provided me though my entire life, specially my time in UnB. Their constant love and support were fundamental in order to finish this dissertation, I love you guys. I must acknowledge my best friend and love, Patricia, without whose love, encouragement and editing assistance (mainly spoken words, not only written), I would not have finished this dissertation. Trust me, you have made me a better person. I would also like to thank Ethel, Aline, Thais, Natalia, Erich, Camila, Carlos, Danusa, Murilo, Dan, Higor, Lores and Ju (do not hate me if I am forgetting someone, please!). For you guys that are always there, I yell "thank you"!

Ana Carolina Cardoso de Sousa

ABSTRACT

Pipelines still are the most efficient, safe, ecological and economical environmental to transport crude oil over long distances. However, the transported oil and the environment in which the pipeline is located may corrode the metal to the point of failure, affecting not only production but also the environment. In addition, activities such as inspection and maintenance are more complex due to difficult access – exposure to toxins, a wide variety of terrains and the special cloths are just some of the challenges. Therefore, pipelines require processes recurrent and autonomous, which motivates the development of new technologies: the machinery of inspection should be cheap, robust and versatile for maintenance, cleaning, removal of fluids, product separation and inspection. The reconfigurable modular robots are autonomous machines with variable morphology and, with the reorganization of the connectivity of parts (called modules), this architecture offers a greater degree of flexibility and fault tolerance at a lower cost. Because of its low cost, robustness and versatility reconfigurable modular robots can perform inspection tasks and reduce production costs in the Oil and Oil Industry. The objective of this work is to design, build and test a module of a reconfigurable modular robot with sensors for inspection in pipelines, called ErekoBot. Each module must have the ability to estimate its own pose, detect an obstacle and align yourself with a plan (simulating a pipe). In this work, the most suitable sensors for ErekoBot were chosen: four infrared sensors and an inertial measurement unit. After the definition of the sensors, the complete module was designed and its prototype was built, considering shape, size, weight, electronic circuit, position of components and material. Tests with the prototype has shown that the module is capable of (1) to estimate its own orientation, (2) detecting the presence of obstacles and (3) align with a plane. These abilities are sufficient to allow a situation where the robot must move moved through a pipeline, avoid obstacles and stop at a specific position to perform an inspection inside the tube.

RESUMO

Oleodutos ainda são os meios mais eficientes, seguros, ecológicos e econômicos para transportar petróleo bruto a longas distâncias. Porém, o petróleo transportado e o meio em que o oleoduto se encontra podem corroer o metal a ponto de surgir falhas, afetando não só a produção, mas também o meio ambiente. Além disso, atividades como inspeção e manutenção são dificultadas devido ao difícil acesso para operadores – exposições a toxinas, uma grande variedade de terrenos e a utilização de roupas específicas são apenas alguns dos desafios. Portanto, oleodutos requerem ainda mais processos recorrentes e autônomos, o que motiva o desenvolvimento de novas tecnologias: o maquinário de inspeção deve ser barato, robusto e versátil para tarefas de manutenção, limpeza, remoção de líquidos, separação de produtos e inspeção. Os robôs modulares reconfiguráveis são

máquinas autônomas com morfologia variável e, com a reorganização das conectividades de suas partes (chamados módulos), essa arquitetura oferece um maior grau de flexibilidade e tolerância a falhas por um custo menor. Por serem baratos, robustos e versáteis, os robôs modulares reconfiguráveis podem realizar tarefas de inspeção e reduzir custos de produção na Indústria do Petróleo e Óleo. O objetivo deste trabalho é projetar, construir e testar um módulo de um robô modular reconfigurável com sensores para inspeção em tubulações, chamado ErekoBot. Cada módulo deve ter a capacidade de estimar sua própria pose, detectar um obstáculo e alinhar-se com um plano (simulando uma tubulação). Neste trabalho foram escolhidos os sensores mais adequados para o ErekoBot: quatro sensores infravermelhos e uma unidade de medição inercial. Depois da definição dos sensores, o módulo completo foi projetado e seu protótipo construído, considerando forma, tamanho, peso, circuito eletrônico, posição dos componentes e material. Os testes com o protótipo mostraram que esse módulo é capaz de (1) estimar sua própria orientação, (2) detectar a presença de obstáculos e (3) alinhar-se com um plano. Essas habilidades são suficientes para simular uma situação em que o robô deve se locomover por uma tubulação, desviar de obstáculos e parar em uma posição específica para realizar uma inspeção no interior do tubo.

Table of Contents

List of Figures

List of Tables

List of Symbols

Nomenclature

O	A signal output,
I	A signal input,
O_{MIN}, O_{MAX}	Minimum and maximum values of O,
I_{MIN}, I_{MAX}	Minimum and maximum values of I,
O_M	Environmental output,
I_M	Environmental input,
m	Mass,
k	Constant stiffness of a spring,
b	Damping factor,
$F_{external}$	External force,
$F_{inertial}$	Inertial force,
$F_{damping}$	Damping force,
F_{spring}	Spring force,
t	Time,
$a(t)$	Acceleration at time t,
$x(t)$	Distance at time t,
ω_r	Resonant frequency,
ζ	Damping ratio,
n	Normal force,
τ	Torque,

g	Gravitational force,
h	Distance,
M	Angular momentum,
M_i,M_f	Initial and final angular momentum,
θ	Angle,
ω_p	Procession motion velocity,
$H(s)$,$G(s)$,$X(s)$,$A(s)$	Transfer functions,
f	Frequency,
I	Identity matrix,
$\hat{z}$	Estimated value of z,
n_1,n_2	Noise,
$R_x(\phi)$	Rotation matrix of x axis by angle ϕ,
$R_y(\theta)$	Rotation matrix of x axis by angle θ,
$R_z(\psi)$	Rotation matrix of z axis by angle ψ,
$\mathbb{H}$	Hamilton domain,
$\mathbb{R}^3$	Three-dimensional vector space,
q	Quaternion,
i,j,k	Imaginary units,
$\bar{q}$	Conjugation of q,
q_v	Vector part of q,
q_r	Real part of q,
u	Unit quaternion,
v	Vector in $\mathbb{R}^3$,
ω	Angular velocity,
L	Length,
ψ	Bending angle,
q^g,q^a	Quaternions from the gyroscope and the accelerometer,
T_s	Sample rate time,

P_{err} Scaling factor,

o_t Orientation vector from accelerometer and gyroscope at time t,

d_t Distance vector from IR sensors at time t,

μ_s Coefficient of friction,

V Output voltage,

Abreviations

2D Two dimensional,

3D Three dimensional,

AC Alternating Current,

ADC Analog-to-digital converter,

CPU Central Processing Unit,

DAC Digital-to-analog converter,

DC Direct Current,

DoF Degrees of Freedom,

FPU Foating-point unit,

I2C Inter-Integrated Circuit,

I/O Input and Output,

IMU Inertial Measurement Unit,

IR Infrared,

LED Light Emissor Diode,

LiPo Lithium Polymer,

LSB Least Significant Bit,

MEMS Micro Eletromechanical Systems technology,

PCB Printed Circuit Board,

PID Proportional Integral Derivative controller,

PIG Pipeline Inspection Gauge,

PWM Pulse Width Modulation,

RF	Radio Frequency,
SMD	Surface-mount device,
SRM	Self Reconfigurable Modular,
USART	Universal Synchronous Asynchronous Receiver Transmitter

Sensores em Robótica Modular para Inspeção em Tubulações: Projeto e Testes do Módulo *ErekoBot* σ

Capítulo 1: Introdução

Apresentação

Este trabalho trata de robôs modulares reconfiguráveis com foco específico no estudo da instrumentação para simples tarefas em inspeção em tubulações. O principal problema enfrentado é como projetar e construir um módulo de um robô modular reconfigurável capaz de se alinhar com as paredes internas de uma tubulação.

Há desafios nas decisões para o projeto de robôs, incluindo qual tipo de sensor utilizar. A Instrumentação é a ciência das medições e do controle de variáveis de processo; suas técnicas são essenciais em todos os ramos da tecnologia e ciência nos dias de hoje (BENTLEY, 2005). Como técnicas de produção modernas determinam limites cada vez menores de precisão com menores custos, há um crescimento nos esforços de pesquisa e desenvolvimento de instrumentos para medir, registrar e controlar as variáveis do processo.

Além da instrumentação, outro importante desafio em desenvolvimento de robôs reside em fornecer versatilidade: capacidade de um robô em executar uma variedade de tarefas em diferentes terrenos. Esse tema é de especial relevância quando o ambiente é desconhecido – como em explorações, operações de resgate e inspeção em ambientes de risco. O ambiente das tubulações na Indústria do Petróleo e o Gás se encaixa nessa situação: é um ambiente industrial perigoso com uma variedade de tarefas em diferentes terrenos. A inspeção em tubulações necessita de sistemas e processos autônomos recorrentes, motivando novas tecnologias, assim como a de robôs modulares reconfiguráveis.

A abordagem tradicional de robôs é primeiro estudar as características do terreno e, em seguida, criar a estrutura mais adequada do robô para essa situação específica. Nessa abordagem, uma escolha de projeto ruim ou a desvalorização de algumas variáveis do ambiente resultam em reformulações do robô.

Em 1994, Mark Yim, em sua tese de doutorado (YIM, 1994), propôs empregar módulos simples conectados um ao outro para criar diferentes configurações no mesmo robô. Assim, os robôs poderiam mudar sua forma, o que lhes permitiria se locomover em diferentes terrenos. Nessa nova proposta, é possível criar diferentes configurações com módulos básicos.

Embora essa abordagem possa parecer motivadora, é importante ressaltar que um robô convencionalmente projetado para uma determinada tarefa será sempre a realizará melhor do que um sistema modular reconfigurável. Robôs modulares reconfiguráveis são uma alternativa para sistemas em que os robôs convencionais não atendem aos requisitos do ambiente. A robustez e versatilidade dos robôs modulares os tornam sistemas redundantes, o que pode cruzar a linha entre convencional e modular. As complexidades mecânicos e computacionais aumentam substancialmente com a modularidade do sistema, como seria de se esperar.

No entanto, um robô modular reconfigurável com sensores é capaz de interagir com o mundo, adaptarem às alterações do ambiente. Este tipo de robô pode: (1) diminuir o custo de fabricação e de manutenção, (2) aumentar o número de aplicações em comparação com os robôs convencionais e (3) aumentar a robustez do sistema.

Objetivos

O principal objetivo deste trabalho é projetar, construir e testar um robô modular reconfigurável com sensores. É desejável que este robô reagir a alterações ambientais a partir de sensores de resposta.

A missão do robô é inspecionar Gathering Lines de tubulações que geralmente têm as seguintes características:

- Eles são usados para o transporte de petróleo bruto ou de gás natural da cabeça do poço para um ponto de coleta central;
- Eles têm tamanho pequeno diâmetro: entre $101,6\,mm$ to $304,8\,mm$;
- Eles trabalham em baixa pressão e fluxo.

Há muitas maneiras de construir este módulo dependendo do orçamento, materiais, tamanho, peso e forma. O módulo aqui projetado é baseado em robôs anteriores desenvolvidos na Universidade de Brasília. Os objetivos concretos estão listados abaixo, cada um ligado a uma pergunta:

1. Descobrir os tipos de sensores comumente empregado em robótica módulos de robôs modulares recentes (*Que tipo de sensores são mais comuns em módulos recentes?*).
2. Determinar as tarefas básicas que o módulo proposto deve executar (*Que tipo de requisitos o módulo possui?*)
3. Analisar que tipo de sensores estão disponíveis no mercado (*Que tipo de sensores vão ser usados neste módulo?*)

4. Depois da escolha dos sensores, ainda é necessário projetar o módulo completo (*Qual a forma, tamanho, peso, circuitos e materiais do módulo?*)

5. Fabricação todo o módulo e testar se os sensores atendem aos requisitos (*Os sensores escolhidos são suficientes para as nossas necessidades?*)

Alguns aspectos relevantes de robôs modulares reconfiguráveis não fazem parte deste trabalho:

1. Locomoção do módulo, porque para analisar a instrumentação não é necessário analisar a locomoção dos módulos.

2. Conexões entre os módulos, uma simples conexão é empregada apenas para garantir que os módulos sejam manualmente reconfigurável.

3. Programação distribuída de alto, uma vez que estes módulos não estão nesse nível de programação ainda.

Estrutura do Documento

O primeiro capítulo contextualizou a dissertação e os objetivos do trabalho. No segundo capítulo, o embasamento teórico necessário para a plena compreensão do trabalho desenvolvido é descrito. O terceiro capítulo apresenta os modelos e conceitos para o módulo e a instrumentação.

O Capítulo 4 combina o conhecimento apresentado nos capítulos anteriores para apresentar o projeto do robô proposto. No quinto capítulo, as experiências mais relevantes são descritas e, finalmente, o sexto capítulo apresenta algumas futuras linhas de investigação.

Capítulo 2: Breve Revisão de Inspeção em Tubulações, Robôs Modulares Auto-reconfiguráveis e Instrumentação

Esse capítulo familiariza o leitor com o backgroud do trabalho. Devido ao fato do trabalho se estende em mais de um campo, são apresentadas as principais questões subjacentes a este projeto: (1) principais descobertas sobre os temas de pesquisa, por quem e quando, (2) os principais pontos de vista em torno dos assuntos investigados e (3) algumas conclusões gerais sobre o estado da arte, incluindo as lacunas que permanecem nas áreas.

A Seção 2.1 resume Inspeções em Tubulações na Indústria do Petróleo e Gás, com suas características e problemas. Além disso, a Seção 2.2 introduz os Robôs Modulares Auto-reconfiguráveis (robôs SRM) com definição, taxonomia, aplicação e exemplos. Com isso em mente, são mostrados na Seção 2.3 alguns sensores e filtros normalmente aplicada na instrumentação de um Robô Modular Auto-recofigurável para inspeção de dutos. Finalmente, a Seção 2.4 mostra como representar rotações espaciais em três dimensões utilizando ângulos de Euler.

Capítulo 3: Projeto Conceitual e Modelamento

Introdução

Este capítulo apresenta os conceitos e modelos empregados no estudo de sensoriamento em robótica modular auto-reconfigurável. Apresenta-se os conceitos do *ErekoBot* σ , em seguida demonstra-se os modelos dos sistema de instrumentação: atitude, distância e filtragem. Com isso, propõem-se o Algoritmo de alinhamento utilizando a resposta dos sensores IR e da IMU.

Modelo Conceitual do *ErekoBot* σ

A simplicidade e robustez caracteriza o conceito do *ErekoBot* σ , que evita altos custos com materiais e processamento. Um módulo é representando como segmentos de linha e essas linhas são conectadas a um eixo de coordenadas 3D que representa o robô modular completo.

A seção apresenta os parâmetros mínimos de projeto, uma possível aplicação e os objetivos do *ErekoBot* σ : (1) detectar um objeto, (2) se alinhar com um plano (simulando uma tubulação) e (3) estimar sua própria atitute.

Alguns parâmetros básicos (como dimensões, tipo de microcontrlador e número de graus de liberdade) foram definidos para começar a projetar o robô.

Modelo de Instrumentação

Quando estimando a pose do robô, é proposto um modelo de atitiude utilizando ângulos de Euler a patir dos resultados do giroscópio e acelerômetro. O modelo do giroscópio é baseado na integração numérica dos resultados e do acelerõmetro, na orientação do corpo relativa a gravidade no eixo de coordenadas de navegação.

Como o giroscópio e o acelerômetro possuem vantagens e desvantagens complementares, um filtro complementar é proposto para combinar esses dados para uma melhor resposta. A estimação da distância ajusta as direções normais da face do *ErekoBot* σ com relação a superfície do objeto detectando o erro de atitute utilizando sensores IR.

Algoritmo de Alinhamento

Então, o algoritmo de alinhamento é proposto para mudar a posição do *ErekoBot* σ enquanto os sensores IR estão desalinhados com o plano.

Capítulo 4: Projeto do Módulo

Este capítulo apresenta todos os passos para a construção do *ErekoBot* σ . O projeto começa pela seleção dos sensores, verificando quais sensores costumam ser utilizados em robótica modular

auto-reconfigurável.

Em seguida, apresenta-se o projeto eletrônico (microcontrolador, comunicação, dispositivos eletrônicos e alimentação externa) e o projeto mecânico (conexão intermodular, servo motor, materiais, descrição geométrica e análise de peso). Com o módulo construido, fizemos uma análise do módulo, comparando-o com outros módulos de robôs modulares auto-reconfiguráveis.

Capítulo 5: Experimentos e Resultados

Este capítulo descreve as plataformas e configurações desenvolvidas para realizar os experimentos. E, então, os resultados são apresentados.

Os experimentos descritos nesse capítulos objetivam verificar a qualidade da informação obtida com os dados dos sensores. Três tipos de experimentos foram realizados: de orientação (dados da IMU), de distância (dados dos sensores IR) e de alinhamento (para posicionamento do módulo paralelo a um obstáculo).

Experimento de Orientação

O principal objetivo desses experimentos foi estimar a posição do *ErekoBot* σ utilizando o acelerômetro e o giroscópio. Nesses experimentos, o microcontrolador lê os dados da IMU e os envia pelo o Digi® XBee® 1mW PCB radio module para o host, que filtra o sinal usando o Filtro complementar.

As leituras da IMU foram testadas em duas diferentes situações: quando o módulo está na posição de repouso e quando o módulo está em movimento. Os experimentos são considerados bem sucedidos quando é possível estimar os ângulos pitch sem o desvio do giroscópio.

Os módulos têm apenas um grau de liberdade, que pode ser disposto de forma a realizar os ângulos de pitch e yaw (ver Figura 5.1). Como não é possível fazer o módulo de executar um movimento de rolagem, esse ângulo não será analisado aqui.

Além disso, por causa do Gimbal Lock (Seção 2.4.1.1), não é possível estimar a posição yaw com o filtro complementar combinando as informações do acelerômetro e giroscópio. Portanto, o único ângulo a ser analisado é o ângulo pitch.

O filtro complementar é ajustado no host e em seguida, o resultado encontrado de P_{err} é utilizado em cada módulo.

Método Experimental

A figura 5.2a mostra a posição e a configuração para o primeiro experimento. O módulo repousa na posição 0° e as medidas da IMU são lidas por 80 segundos, com um tempo de amostra de 200 ms. Portanto, o módulo envir 400 medições para o host, que calcula off-line os sinais filtrados com o algoritmo do filtro Complementar.

A figura 5.2b mostra como o módulo se movimenta da posição $-45°$ a $45°$. O algoritmo utilizado pelo módulo é constituído pelos seguintes passos:

1. Ler a IMU 10 vezes;
2. Enviar as medições para o host;
3. Esperar $200\,ms$;
4. Avançar $5°$;
5. Se a posição for inferior a $45°$, voltar para o passo 1.

Resultados

Usando as equações (3.2), (3.12), (3.13), o host calcula o ângulo pitch a partir das medições da IMU, enquanto o módulo permanece na posição de repouso ($0°$), conforme indicado na Figura 5.3. É possível observar que os os resultados baseados no acelerômetro forneceram valores de ângulos exatos: mesmo que o valor médio varie no tempo, a média é $-3.30°$ com um desvio padrão de $12.74°$. Esses valores não são precisos, porque há uma grande variação em cada medição. Em contraste, os ângulos baseados giroscópio foram precisos, mas inexatos devido ao desvio que ocorre ao longo do tempo.

Os resultados sugerem que os dados do acelerômetro e giroscópio são complementares, reafirmando a possibilidade do uso do Filtro Complementar. Os resultados da estimativa de ângulo usando o filtro Complementar são apresentados na Figura 5.4. Como mencionado na Seção 3.3.1.3, quando P_{err} é próximo de 0, o Filtro Complementar elimina as leituras do acelerômetro (linhas mais escuras) e quando P_{err} está perto de 1, o Filtro Complementar não leva em conta as medidas do giroscópio.

O ângulo pitch também foi calculado pelo host a partir das medições da IMU quando o motor está se movimentando de $-45°$ a $45°$, e as leituras são mostrados no gráfico da Figura 5.5. Esses resultados são semelhantes aos do gráfico da Figura 5.3: o acelerômetro fornece ângulos exatos, mas imprecisas, enquanto os ângulos do giroscópio são precisos, mas inexatos. A figura 5.6 apresenta o gráfico com a estimativa obtida usando o filtro Complementar, quando P_{err} é 0.9, 0.7, 0.5, 0.3 e 0.1.

Os resultados das Figuras 5.4 e 5.6 fornecem dados suficientes para empregar o Filtro Complementar para estimativas futuras, com with $P_{err} = 0.5$, porque é o resultado que mais se aproxima da linha reta entre $-45°$ a $45°$.

Experimento de Distância

O principal objetivo desses experimentos é detectar a presença de um obstáculo e estimar sua distância da face do *ErekoBot* σ onde os sensores IR são fixados. A dependência entre a saída, tensão, e a distância estimada para os sensores IR não é linear, o que sugere a necessidade de

calibrar e linearizar o Sensor IR SHARP GP2Y0A41SK0F (Seção 2.3.1). Após a calibração, uma lookup table é gerado e é inserida no microcontrolador para ser utilizado enquando se estima a distância entre a face do módulo e um obstáculo no meio.

O experimento é bem sucedido quando o *ErekoBot* σ é capaz de detectar um obstáculo e estimar sua distância usando a tabela de referência.

Método Experimental

O sensor IR SHARP GP2Y0A41SK0F detecta obstáculos e mede as medidas dentro de uma faixa entre 40 a 300 mm. Com o objetivo de garantir a detecção e medição corretas, o Sensor IR foi calibrado através da análise da curva de tensão não-linear detectar um obstáculo cinza entre 10 to 350 mm com uma resolução de 10 mm. A Figura 5.7 mostra a disposição experimento.

A fim de considerar a possíveis mudanças por causa da histerese e das condições do meio, as medidas foram realizadas em ambas as direções. Inicialmente, o sensor está localizado na posição 10 mm, e então o sensor IR se move lentamente para outras medidas até atingir 350 mm. Nesse ponto, o sensor é movido para trás e as medições são feitas novamente. Para certificar-se os resultados são redundantes, as medições são repetidos quatro vezes para cada posição.

Uma curva com base nas leituras dos sensores é traçada, e a faixa aceitável é determinada. A melhor equação exponencial é escolhida com a ajuda do aplicativo Curve Fitting Toolbox® do MATLAB®.

Resultados

A Figura 5.8 mostra a curva obtida pelos valores médios. A dependência entre a saída de tensão e distância de proximidade para sensores de IR é não-linear e bastante semelhante à curva do datasheet (SHARP, 2002). A partir desta curva, é possível especificar um intervalo de trabalho entre 5 e 28 cm .

Com o Curve Fitting Toolbox® do MATLAB®, a melhor equação foi encontrada.

A fim de construir a tabela de referência, necessária para calcular a distância baseado na medição de tensão, a Equação (5.2.2) é invertida.

Experimento de Alinhamento

O objetivo principal desse experimento é detectar a presença de um obstáculo, estimar sua distância da face do *ErekoBot* σ e alinhar a superfície do módulo com a superfície do obstáculo. A fim de verificar o algoritmo de alinhamento da Seção 3.4, o algoritmo foi testado com diferentes planos em diferentes distâncias.

Método Experimental

Combinando os resultados das Seções 5.1.2 e 5.2.2, o seguinte algoritmo de alinhamento foi aplicado:

1. O *ErekoBot* σ lê a orientação da haste no tempo t ($orient_t(6)$) e calcula o ângulo pitch através de um filtro complementar com $P_{err} = 0.5$;
2. O *ErekoBot* σ lê a distância de objetos em tempo de t ($proxim_t(4)$), utilizando a tabela de referência a partir da Equação (5.1);
3. Através do Digi® XBee® 1mW PCB radio module , o módulo envia $orient_t(6)$, $proxim_t(4)$ e t para o host;
4. Se as distâncias são diferentes, o servo motor gira para um ângulo θ proporcionalmente a diferença medida pelos sensores IR, a fim de alinhar com o plano;
5. Após a adicionar 1 a t e esperar por $300\,ms$, o *ErekoBot* σ retorna ao passo 1.

O alinhamento foi testado com três superfícies diferentes: uma folha de papel branco, um papel cartão cinza e uma folha de papel preta em diferentes posições. O objetivo é testar se o alinhamento trabalha com diferentes cores (branco, cinza e preto) e estimar quanto tempo leva para o *ErekoBot* σ para alinhar com os planos.

Resultados

Figuras 5.13, 5.14 e 5.15 apresentaram os experimentos de alinhamento. Os resultados foram semelhantes com as três superfícies diferentes testados: o *ErekoBot* σ se alinha com o plano. Superfícies brancas refletem melhor do que superfícies pretas, e, como sensores IR funciona com luz infravermelha refletida (Seção 2.3.3), o plano branco funciona mais rapidamente que os planos cinza e preto.

Considerando resultados quantitativos, os três ensaios são apresentados. O *ErekoBot* σ executa o algoritmo de alinhamento em uma superfície cinza, e as diferenças entre as distâncias medidas dos sensores IR e as posições de orientação da IMU (após a execução de um filtro complementar) são enviados para o host. Os testes foram realizados dez vezes em cada posição.

No primeiro teste, o *ErekoBot* σ -se alinhou-se com o plano na posição $15°$ em menos de 1 segundo, como mostrado nas Figuras 5.10a e 5.10b. O servo motor gira no sentido dos ângulos positivos até a haste do módulo se alinhar ao ângulo de $16°$.

O mesmo acontece no segundo teste, o *ErekoBot* σ alinha-se com o plano na posição $-25°$ em menos de 1 segundo, como mostrado nas Figuras 5.11a e 5.11b. O servo motor gira no sentido dos ângulos negativos até o ângulo da haste convergir para $27°$.

No terceiro e último teste, o *ErekoBot* σ alcança o alinhamento com o plano na posição $-40°$ em menos de 1.2 segundos, como mostrado nas Figuras 5.12a e 5.12a. O servo motor gira no sentido dos ângulos negativos até que o valor $-43°$ seja atingido.

Os testes apontaram algumas limitações:

- Luz: quando há uma luz apontada diretamente para um sensor IR, as respostas de medição são afetadas. O sensor IR considera que existe um objeto e o alinhamento não funciona corretamente.
- Distâncias superior a $280\,mm$: como esperado, o sensor IR não mede adequadamente distâncias maiores do que o máximo da faixa. O alvo é considerado a uma distância de $280\,mm$ para qualquer distância maior.
- Distâncias menores que $50\,mm$: como esperado, o sensor IR não mede adequadamente distâncias menores do que o mínimo do intervalo. Observando o gráfico da Figura 5.8, é possível que o *ErekoBot* σ considere o objeto a uma posição dentro da faixa aceitável de valores $50\,mm$ a $280\,mm$. Para trabalhos futuros, um algoritmo que considere estatísticas baseadas em distâncias anteriores podem ser desenvolvidos para trabalhar nesses casos.
- Posições do servo motor maiores que $35°$ e menores que $145°$: a fim de evitar a colisão das peças internas módulo, foram determinadas posições máximas e mínimas ao servo motor.
- A calibração do sensor IR: a Figura 5.9 funciona apenas para obstáculos cinzas, cores diferentes podem produzir diferentes lookup tables. No entanto, como o *ErekoBot* σ se alinha com base nas diferenças entre essas distâncias, se o alinhamento ocorre em uma superfície homogênea, (mesma cor e textura), os resultados não são afetados.

Considerando estas limitações não é possível utilizar esse robô em tubulações ainda.

Conclusões

Um módulo SRM foi projetado e fabricado respeitando o desenho conceptual proposto no Capítulo 4, o que nos permitiu testar os modelos do Capítulo 3. Os sensores incorporados são usados para:

1. Estimando a orientação do *ErekoBot* σ usando o Filtro Complementar.
2. Calibrar os sensores IR para medições de distância.
3. Alinhar a face do *ErekoBot* σ com um plano.

Com os resultados dos testes de IMU (Figura 5.3 e Figura 5.5), é possível verificar que as medições para estimar a orientação módulo são ruidosas ou desviam ao longo do tempo. Para superar este problema, um filtro complementar foi empregado, o que diminui o ruído e o desvio,

estimando rapidamente e não sobrecarregando o processador do módulo. Portanto, para efeitos de estabilização mais rápida e mais precisa do *ErekoBot* σ , o filtro complementar é escolhido como padrão utilizado para esse tipo de processos com a IMU.

A calibração do sensor IR forneceu uma curva semelhante à que aparece no datasheet (SHARP, 2002), e os resultados da equação (5.1) foram convertidos em uma lookup table programado no Atmel® AVR® ATmega8 .

Com estes resultados, foi possível realizar o alinhamento, e demonstrar a viabilidade do algoritmo descrito na Seção 3.4. Mas não é possível confirmar que o robô não pode ser utilizado para aplicações de inspecção oleoduto ainda.

Capítulo 6: Conclusões

Este capítulo apresenta as principais contribuições desta dissertação com uma breve discussão sobre as suas implicações. E, em seguida, indica algumas das possíveis futuras linhas de investigação para este trabalho.

Principais contribuições

O *ErekoBot* σ é um módulo reconfigurável simples com sensores capaz de reagir ao ambiente. Neste trabalho, o módulo foi projetado, fabricado e testado, o que nos permitiu responder às questões apresentadas na Seção 1.2:

1. *Que tipo de sensores são mais comuns em módulos recentes?* A Tabela 4.2 retoma as principais características dos cinco dos módulos mais recentes. Os sistemas Roombots e SMORES não tem sensores internos; O Transmote só usa sensores de proximidade; já o Cosmo e UBot têm uma seleção mais completa de sensores de orientação e de proximidade.

2. *Que tipo de requisitos o módulo possui?* A seção 3.2 apresenta os três principais requisitos para o *ErekoBot* σ : as habilidades para (1) detectar um obstáculo (Figura 3.3a), (2) se alinhar com um plano simulando uma tubulação (Figura 3.3b) e (3) estimar a sua própria posição (Figura 3.3c).

3. *Que tipo de sensores vão ser usados neste módulo?* O *ErekoBot* σ possui quatro sensores infravermelhos (GP2Y0A41SK0F) para detecção de obstáculos e o algoritmo de alinhamento. Além disso, *ErekoBot* σ possui uma IMU (com ADXL345 e ITG-3200) para medições de movimento de rotação.

4. *Qual a forma, tamanho, peso, circuitos e materiais do módulo?* O Capítulo 4 foca no projeto do módulo, e as figuras 4.6, 4.9, 4.10 e a Tabela 4.2 ilustram e retomam a forma, tamanho, peso, circuitos e materiais do módulo completo.

5. *Os sensores escolhidos são suficientes para as nossas necessidades?* O Capítulo 5 apresenta os resultados dos experimentos. O *ErekoBot* σ é capaz de: (1) detectar um obstáculo entre

5 e 28 *cm* (Figura 5.9), (2) se alinhar com um plano que simula uma tubulação (Figuras 5.13, 5.14 e 5.15), e (3) estimar a sua própria pose com um Filtro Complementar do giroscópio e acelerômetro medições (Figuras 5.4 e 5.6).

O módulo mostra evidências para a sua aplicação nas inspecções de oleodutos, porém as soluções propostas têm suas limitações e que ainda não são viáveis para aplicação em campo.

Trabalhos Futuros

Algumas linhas de investigação são apresentados para dar continuidade a esta dissertação (a ordem que aparece é uma sugestão da autora de importância):

- *Módulo mais leve* Ainda possível diminuir o peso do módulo, a alterando o acrílico por um material mais leve (tais como resinas, fibras de carbono) e mudando a placa de circuito impresso (PCB) para um dispositivo de montagem superficial (SMD).
- *Cabo/Energia* Diminuir o peso do módulo permite a utilização de uma fonte de energia autonoma, eliminando a necessidade do cabo.
- *Magnetômetro* Para projetos futuros, o magnetômetro pode ser usado para determinar o yaw.
- *Alinhamento e algoritmos de desvio de obstáculos* Este trabalho apresenta códigos e algoritmos para a leitura dos valores de sensores e convertê-los para o sistema mais familiar aos usuários (distância em milímetros, orientação em vetores). No entanto, as leis de controle não foram adicionados a encontrar a melhor resposta para os módulos. É possível fazer um controle de malha fechada do sistema.
- *Conexão* Outra linha de investigação diz respeito a conexão entre os módulos, é fundamental para substituir o Velcro$^{\text{TM}}$ por um sistema macho-fêmea com motores de corrente contínua em cada face do módulo.
- *Métodos de Visão Computacional* A homogeneidade de *ErekoBot* σ possui a vantagem para sua produção em série. No entanto, um dos módulos pode ser diferente e transportar uma câmera para permitir a utilização de métodos de visão computacional para a detecção e alinhamentos obstáculos.
- *Sistemas de Robótica Distribuida* Depois de resolver o sistema de alinhamento, é possível trabalhar com sistemas robóticos distribuídos, analisando quando se conectar, como ligar e que tipo de colaborações entre o robô e os módulos seram estabelecidas.

Embora existam várias limitações neste sistema, o projeto e testes contribuem para a robótica modular reconfigurável a fim de construir um sistema versátil que pode alterar a automação dos processos de manutenção e de detecção de vazamentos no futuro.

Chapter 1

Introduction

1.1 Presentation

This work deals with reconfigurable modular robots with specific focus on the study of instrumentation for simple tasks in pipelines inspection. The main problem faced here is how to design and manufacture a reconfigurable modular robot able to align with the internal walls of a pipeline.

There are challenges in the decisions for robots design, including what type of sensors to use. Instrumentation is the science of measurements and control of process variables; its techniques are essential in every branch of technology and science nowadays (BENTLEY, 2005). As modern production techniques dictates tighter accuracy limits with lower production costs, there is a growth in the research and development efforts of instruments to measure, record and control process variables.

Besides instrumentation, other important challenge in developing a robot resides in providing versatility: the ability for a robot to perform a variety of tasks in different terrains. This is of a special relevance when the environment is unknown - such as outdoors exploration, rescue operations and inspection in hazard environments. The pipeline environment in Oil and Gas industry fits: it is a hazardous industrial environment with a variety of tasks in different terrains. Pipeline inspection needs recurrent autonomous processes and systems, motivating new technologies, such as reconfigurable modular robots.

The traditional approach in robots is to first study the characteristics of the terrain and then design the most adequate structure of the robot for that specific situation. In this approach, a bad design choice or a misevaluation of some environment variable results in the need to redesign the robot.

In 1994, Mark Yim, in his doctoral thesis (YIM, 1994), proposed to employ simple modules joined one to another to make up different configurations in a robot. Thus, the robots can change their form, enabling them to move thought unknown terrains. In this new idea, it is possible to create different configurations with basic modules.

Although this approach may seem motivating, it is important to emphasize that a robot con-

ventionally designed for a particular task will always perform better than a modular system. Reconfigurable Modular Robots are an alternative for systems where the conventional robots do not meet the environment requirements. The robustness and versatility of modular robots make them redundant systems, which may cross the line between conventional and modular. Mechanical and computational complexities substantially increase with the system modularity, as should be expected.

Nevertheless, a reconfigurable modular robot with sensors is able to interact with the world, adapting to changes in the environment. This type of robot may: (1) decrease manufacturing and maintenance cost, (2) increase the number of applications compared to conventional robots and (3) increase robustness.

1.2 Aims

The main aim of this work is to design, manufacture and test a reconfigurable modular robot with sensors. It is desired that this robot react to environmental changes based on sensors response.

The mission of the robot is to inspect Gathering Lines of pipelines that usually have these characteristics:

- They are used to transport crude oil or natural gas from the wellhead to a central collection point;
- They have small diameter size: between 101.6 mm to 304.8 mm;
- They work on low pressure and flow.

There are many ways to construct this module depending on budget, materials, size, weight and shape. The module here designed is based on previous robots developed at University of Brasília. The concrete aims are listed below, each one linked to a question:

1. Discover types of sensors commonly employed in recent SRM robotics (*What kind of sensors are more common in recent SRM robotics?*).
2. Determine the basic tasks that the proposed module must perform (*What kind of requirements does the robot have?*)
3. Analyze what kind of sensors are available to measure variables for the module requirements (*What kind of sensors are going to be used in this module?*)
4. After choosing the sensors, it is still necessary to design the complete module (*What is the shape, size, weight, circuits and materials of the module?*)
5. Manufacture the entire module and test if the sensors meets the requirements (*Does the chosen sensors are sufficient for our needs?*)

Some relevant aspects of SRM are not part of this work:

1. Locomotion of the module,because to verify the instrumentation it is not necessary to analyze the locomotion of the modules.
2. Connections between modules, a simple connection is employed just to make sure the modules are manually reconfigurable.
3. High level distributive programming, as these modules are not at this level of programming yet.

1.3 Structure of the Document

This first chapter introduced the context of the dissertation and the aims of the work. In the second chapter, the theoretical background necessary for the full understanding of the work developed is described. The third chapter presents the models and concepts for the module and instrumentation.

Chapter 4 combines the knowledge showed in previous chapters to presents the design of the proposed robot. In the fifth chapter, the most relevant experiments are described and, finally, the sixth chapter lists some future lines of investigation.

Chapter 2

Brief Review of Pipeline Inspection, SRM Robots and Instrumentation

This Chapter familiarizes the reader with this work background. Due to the fact that our work spans more than one field, we will present the key issues underlying this project: (1) major findings on the research topics, by whom and when, (2) main points of view surrounding the issues investigated and (3) some general conclusions about the state of the art, including gaps that remains in the area.

Section 2.1 resumes Pipeline Inspections in Oil and Gas Industry, with its characteristics and problems. In addition, Section 2.2 introduces Self-Reconfigurable Modular Robots (SRM robots) with its definition, taxonomy, application and examples. With that in mind, we will show in Section 2.3 some sensors and filters usually applied for instrumentation in a Self-Reconfigurable Modular Robot for pipeline inspection. Finally, Section 2.4 shows how to represent spatial rotation in three dimensions using Euler angles.

2.1 Pipeline Inspection

Oil and gas pipelines play a critical role in the transportation process, the World Factbook from the Central Intelligence Agency (Central Intelligence Agency, 2013) estimates $2,225,032$ kilometers of pipelines in the United States, $259,913$ in Russia and $100,000$ in Canada. The lengths for transporting products in Brazil is 251 km for condensate/gas; $17,312$ km for gas; 352 km for liquid petroleum gas; $4,831$ km for oil and $4,722$ km for refined products, totalizing $27,477$ km of pipelines.

In addition to a substantial cost advantage, pipelines also result in fewer leak incidents and personal injuries than other types of transportation (GREEN; FURCHTGOTT-ROTH, 2013), making them the most ecological friendly and economically viable form for transportation of crude oil over long distances. However, there is still a potential leak: the transported oil and the environment may still corrode the pipelines metal to the point of failure, affecting not only production but also

the environment (GREEN; FURCHTGOTT-ROTH, 2013).

Moreover, these hazardous industrial environments are difficult and dangerous places for personnel: exposure to toxins, explosive gases, and bulky personal protective equipment are just a few of the challenges of working in these facilities. Therefore, pipes increasingly require recurrent autonomous processes and systems, which motivates the development of new technologies (ARCHILA; BECKER, 2013). In this scenario, pipeline inspection machines, often robots, must be low-cost, robust and versatile to perform tasks, such as maintenance, cleaning, removal of liquids, separation of products and inspection.

2.1.1 Classification of Pipeline Robots

In the context of pipeline robots, there is a wide variety of tasks – like inspection, maintenance and construction. In 1999, Hirose et al (HIROSE et al., 1999) presented a classification based on locomotion mechanisms and, in 2007, Choi et al (CHOI; RYEW, 2002) improved adding two different types of locomotion, as shown in Figure 2.1.

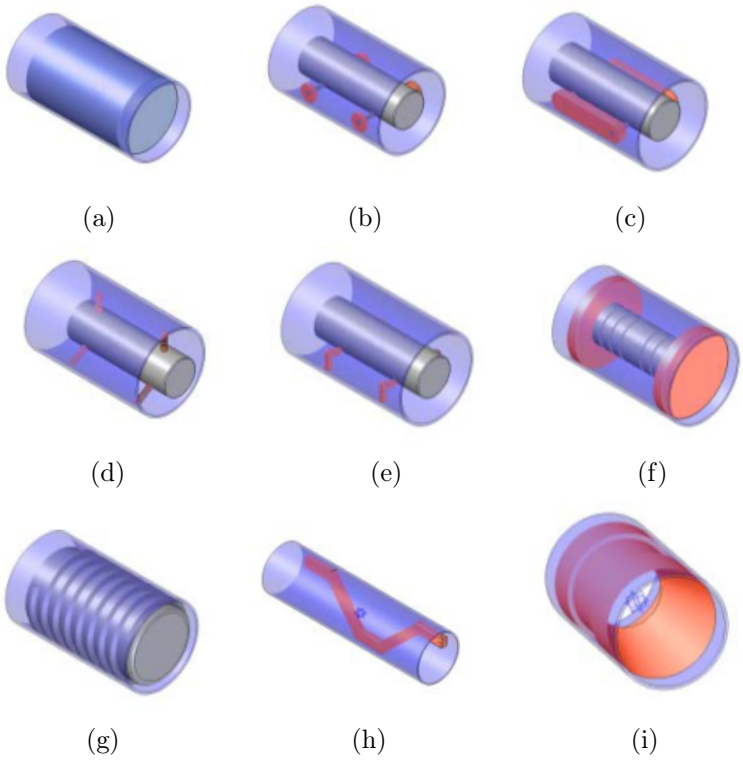
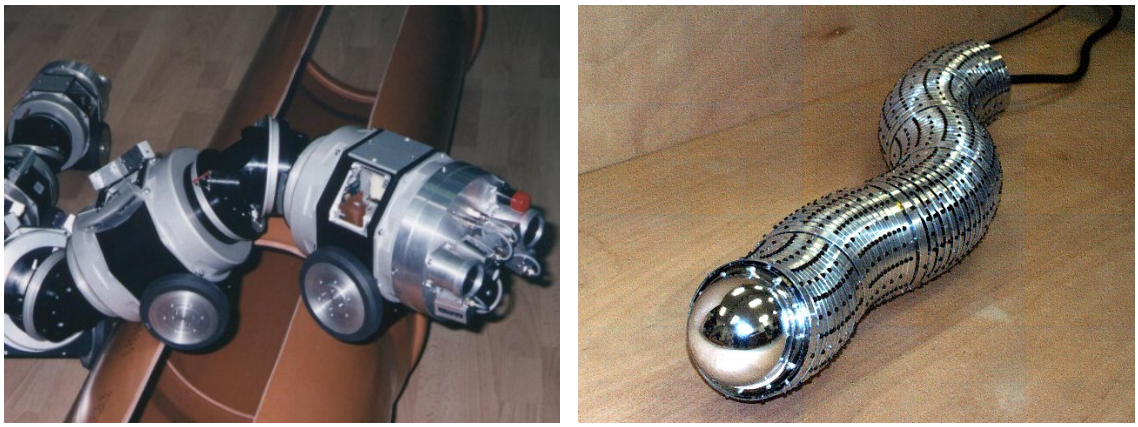


Figure 2.1: Classification of pipeline robots (Figures from (ARCHILA; BECKER, 2013)): (a) PIG. (b) Wheel. (c) Caterpillar. (d) Wall-press. (e) Walking. (f) Inchworm. (g) Screw. (h) Snake. (i) Variable Velocity PIG.

Figure 2.1 presents the classification proposed in (CHOI; RYEW, 2002): (a) The pipeline inspection gauge (PIG), formed by large pieces of machinery pulled together with powerful technology, where the flow-line service generates locomotion for pipelines with large diameters (OKAMOTO et al., 1999); (b) The wheel type, similar to plain mobile robots (HIROSE et al., 1999); (c) The

caterpillar or crawler type, such as the robot Versatrax (INUKTUM,), that uses caterpillars instead of wheels; (d) The wall-press type, that puts pressure on the wall of the duct, and thus generate the necessary traction (CHOI; RYEW, 2002); (e) The walking type, that possess legs producing highly sophisticated motions (PFEIFFER; ROSSMANN; LOFFLER, 2000); (f) The inchworm type, used in small diameters pipelines (BAUER, 2011); (g) The screw or helical type, displaying motion of a screw when it advances in the pipelines (HORODINCA et al., 2002); (h) The snake type, similar to a snake or a annelid animal in nature, like MAKRO (STREICH; ADRIA, 2004) and Slim Slim Robot (OHNO; HIROSE, 2001); (i) And, finally, the variable velocity PIG (Torres Jr; MANZAK; MILLER, 2002), where we can control the PIG speed introducing a bypass through the PIG body.

In this work, we will focus on robots similar to the snake MAKRO (Figure 2.2a) and Slim Slim Robot (Figure 2.2b), and classified as the snake type.



(a) MAKRO (STREICH; ADRIA, 2004). (b) Slim Slime Robot (OHNO; HIROSE, 2001).

Figure 2.2: Examples of Pipeline Robots.

2.2 Self-Reconfigurable Modular Robots

In order to adapt for different terrains and tasks, Mark Yim developed in 1994 (YIM, 1994) the concept of a robot system with variable morphology, nowadays called Self-Reconfigurable Modular Robots (SRM Robots). Besides presenting other robots characteristics (like actuators, sensors and control systems), SRM Robots are also capable of changing its own shape and reorganizing its parts, often called *modules*. Modules are devices with processors, batteries, actuators and special mechanisms like hooks and cameras; they may transmit force, momentum, energy and data for the rest of the robot.

In manufacturing, *modularity* refers to the use of exchangeable parts or options in the fabrication of an object based on standardized units or dimensions, that easies assembly and repair (NORBERG, 2001). *Reconfigurability* is the ability to repeatedly change and rearrange the components of a system in a cost-effective way (SETCHI; LAGOS, 2004). Both these characteristics in SRM robots give them the potential to excel conventional robots in multi-functionality, flexibility,

and robustness, with possibility of morphogenesis (Self-Assembly), self-repair, self-reproduction, scalability and motion generation (MURATA; KUROKAWA, 2007).

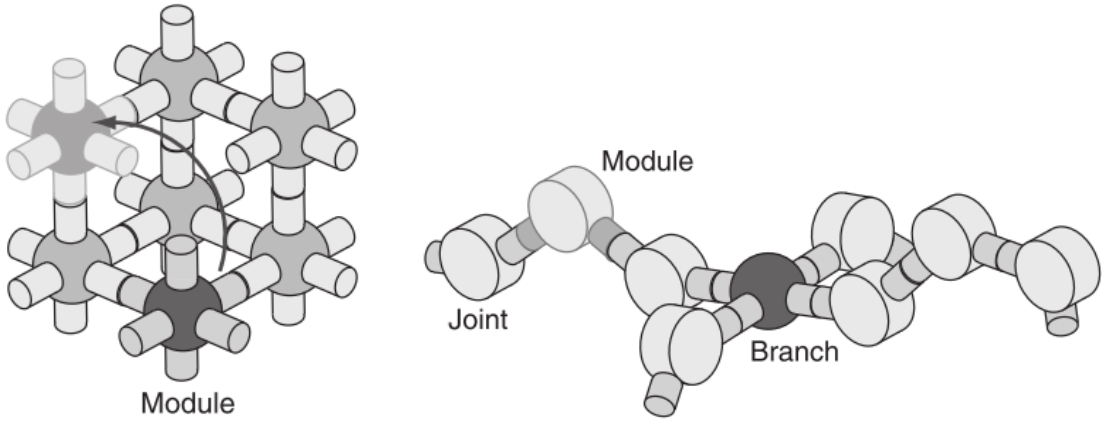
2.2.1 Taxonomy

Generally, we classify SRM Robotic Systems according to their geometric arrangement (MURATA; KUROKAWA, 2007):

Lattice Architecture modules dock into interfaces at points into virtual cells of some regular grid (Figure 2.3a). Usually, few modules are sufficient to accomplish a reconfiguration step with a simpler mechanical design and a simpler computational representation. It is also easier to scale the reconfiguration planning to complex systems.

Chain Architecture modules may reach any point in space and therefore, the robot is more versatile (Figure 2.3b). However, the robot may need more modules to reach a point, making it usually more difficult to accomplish a reconfiguration step. They are more computationally difficult to represent and analyze.

Hybrid Architecture it is a hybrid between the previous architectures: the control and mechanisms are designed in lattice, and the robot is constructed to reach any point in space.



(a) Lattice type. (b) Chain type.

Figure 2.3: Types of SRM robots (Figure from (MURATA; KUROKAWA, 2007)).

We can also classify the SRM robots depending on the way by which units are reconfigured (MURATA; KUROKAWA, 2007):

Deterministic Reconfiguration the location of each module is regulated at all times during reconfiguration with sophisticated feedback control. Therefore, it is possible to guarantee reconfiguration time, even if precise feedback manipulation is necessary.

Stochastic Reconfiguration statistical processes are responsible for modules moving around. The exact location of each module is unknown, except when they are connected to the main structure.

Furthermore, we can classify the modules according to their design (MURATA; KUROKAWA, 2007):

Homogeneous when SRM robot modules have the same design forming a structure suitable to perform the required task. It is simpler to scale it in size, but the functionality is more limited, as more modules are necessary to achieve a given function.

Heterogeneous when SRM robot modules are different, with specialized functions. This type of robot is more compact, and it is easier to add units, but it is also more complex to design, manufacture and simulate.

2.2.2 Brief History of SRM Robotics

The idea of building systems by assembling homogeneous components is not new. It dates back to von Neumann's "Theory of self-reproducing cellular automata" (BANKS, 1970), which assumed complete homogeneity of initial modules. This is an analogy to biological systems, in which cell division from a single cell creates all the differentiated cells. However, biological systems need a vast number of cells to form their bodies and artificial systems need to admit heterogeneity among components or use complex modules with more functions within themselves (MURATA; KUROKAWA, 2007).

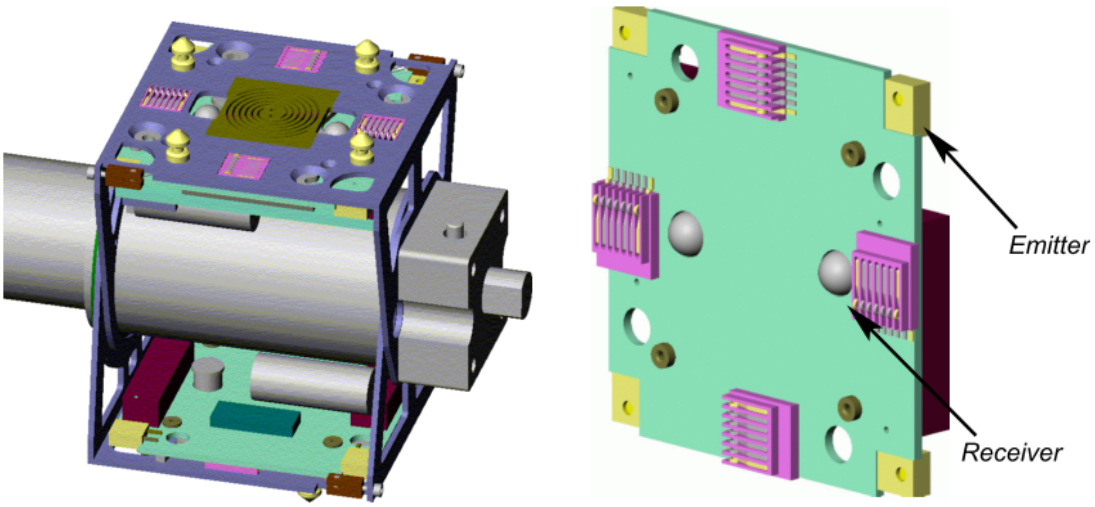
CEBOT (FUKUDA; KAWAUCHI, 1990) was the first SRM robot designed based on heterogeneous modules, such as transportation, rotation joint, telescopic arm, and grasping modules. These combinations made it possible for a CEBOT to perform a variety of tasks.

Polybot (Figure 2.4a) is a system used to highlight methods and issues in the docking requirements of self-reconfiguration (Figure 2.4a). An IR ranging system permits a docking controlled by a closed loop control system: when the modules are close enough, the IR sensors response give a empirically centralized feedback for the docking control (Figure 2.4b). This method proved to be more robust than a computed offset method (YIM et al., 2002).

Several SRM robots with homogeneous modules followed the CEBOT project (MURATA et al., 2002; JORGENSEN; OSTERGAARD; LUND, 2004; YIM et al., 2002). In the homogeneous system, a set of common rules must sufficiently describe the differentiation and behavior of each module when it is part of a specific robot configuration. Therefore, we can replace any module with another one, which simplifies self-repair and self-reproduction schemes.

2.2.3 Recent SRM Robots

This Section shows some of the most recent work on SMR, developed by different research centers around the world, like the Karlsruhe Institute of Technology (KIT - Germany), the École Polytechnique Fédérale de Lausanne (EPFL - Swiss), the Southeast University (SEU - China), the Harbin Institute of Technology (HIT - China) and the University of Pennsylvania (Penn - EUA).



(a) CAD rendering of a PolyBot G2 module (Figure from (YIM et al., 2002)).

(b) IR sensing device on a PolyBot faceplate (Figure from (YIM et al., 2002)).

Figure 2.4: Examples of Pipeline Robots.

2.2.3.1 SMORES

The SMORES (Self- assembling Modular Robot for Extreme Shape-shifting, shown in Figure 2.5) (DAVEY; KWOK; YIM, 2012) is a $100 \times 100 \times 90\,mm$ module designed by the School of Engineering and Applied Science at University of Pennsylvania (U.S.A.). The SMORES module weights $0.52\,kg$ and contains five identical motors to achieve a module with four Degrees of Freedom (DoF)[1]. Power, microcontroller and communications are embedded on the module, but there are no environmental sensing devices on-board yet.

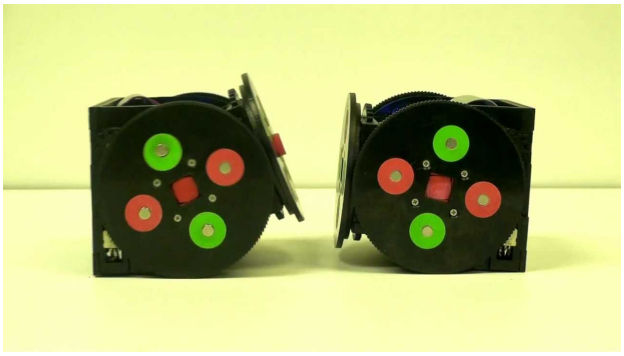


Figure 2.5: SMORES (DAVEY; KWOK; YIM, 2012).

Future revisions expect vision capabilities, local communication between the docking ports of neighboring modules and several environmental sensing abilities, such as torque feedback (current sensing), tactile sensing and orientation sensing to perform self-balance on its two outside wheels.

[1]The number of independent parameters that defines its configuration.

2.2.3.2 UBot

The State Key Laboratory of Robotics and Systems at the Harbin Institute of Technology (China) designed the UBot (CUI et al., 2012; WANG; ZHU; ZHAO, 2013; ZHU et al., 2013) (shown in Figure 2.6), with two 80 mm cube shape. Each module has two independent rotational DoF and weights 280 g; they are also classified as passive or active module, depending on their connecting mechanisms. The universal joints of UBot makes it more flexible for locomotion and reconfiguration (CUI et al., 2012).

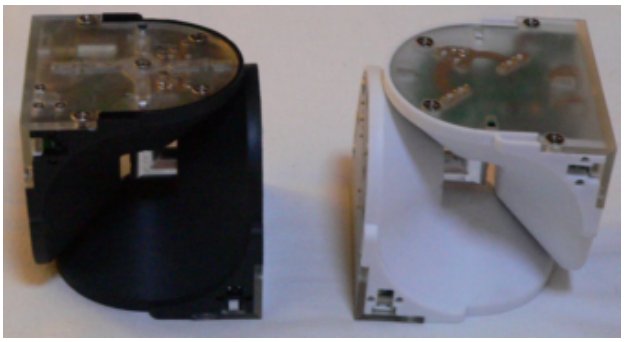


Figure 2.6: UBot (CUI et al., 2012; WANG; ZHU; ZHAO, 2013; ZHU et al., 2013).

The system also have a sensory module with the same cubic structure and dimensions including four types of sensors: wireless vision sensor, accelerometer, infrared range finder, and linear Hall Sensor. A chain module group with a sensory module as the head of the robot docks with a target active module. Figure 2.7 shows the four symmetrically distributed yellow round tags for visual recognition in the target (ZHU et al., 2013).

Figure 2.7: UBot target (ZHU et al., 2013).

The docking approach consists of two phases: pre-positioning preparation and precise positioning (ZHU et al., 2013). In the phase of pre-positioning preparation, the sensory module obtains visual information, extracts the features and generates the motion adjustment parameters through computation in the host PC. Then, the motion module group repeats the process of parameters

adjustment to achieve pre-positioning.

After pre-positioning, the linear Hall sensors capture range information and the host calculates the motion ajustment parameters for the docking algorithm.

2.2.3.3 CoSMO

CoSMO (Collective Self-reconfigurable Modular Organism, shown in Figure 2.8) (LIEDKE et al., 2013) consists of several uniform building blocks, the CoSMO modules, and was designed by the Institute for Process Control and Robotics, Karlsruhe Institute of Technology (Germany). The module boundary box is an exact cube with dimensions $105 \times 105 \times 105\,mm$, weight $1.25\,kg$ and one DoF. Embedded in the module, there is a battery pack and a PCB with a Micro Controller Unit, which scans and preprocesses sensor data.

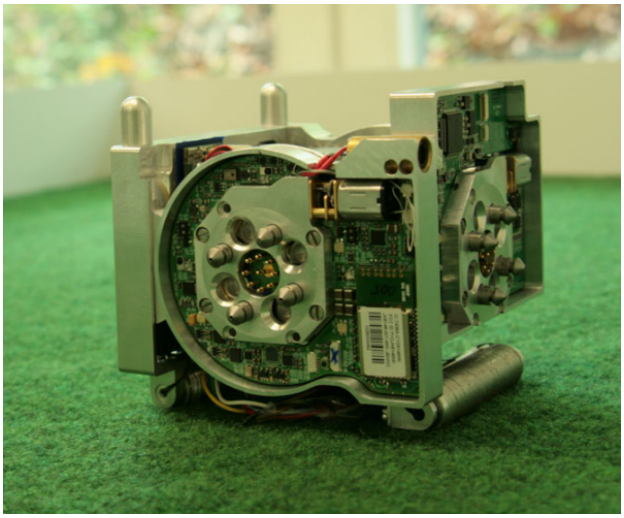


Figure 2.8: CoSMO (LIEDKE et al., 2013).

The system also supports proprioceptive sensors (such as Hall sensor, gyroscope, accelerometer and motor current sensor) and exteroceptive sensors (such as camera, infrared sensor and microphone). For docking, one DC gear motor actuates on the hooks for grasping four bolts of other robots docking units. Based on the infrared sensors readings, the docking approach algorithms move the robot exactly as required by the mechanics of the docking unit. In the future, force sensitive resistors based sensors will aid docking units to measure forces and torques between docked robots.

2.2.3.4 Roombots

The Roombots system (MOECKEL et al., 2013; BONARDI et al., 2013; VESPIGNANI et al., 2013; SPRöWITZ et al., 2014) (shown in Figure 2.9), designed by the Biorobotics Laboratory at École Polytechnique Fédérale de Lausanne (Switzerland), focus on building blocks for adaptive pieces of furniture able to move, self-assemble and self-reconfigure. Two half-spheres linked together with revolute joints forms a $110\,mm$ diameter module weighting $1.4\,kg$. Each Roombot has three

DoF and is a fully autonomous robot with a 4-cell LiPo battery with autonomy for about 1 hour.

The user control and configure Roombot modules from a PC using a wireless Bluetooth communication module. In addition, slip rings allow the transmission of communication and power within the modules. In open-loop experiments, there is a PID to controle the position of RB modules through relative position sensors at each joint. However, there are no additional sensors for sensing the alignment of a module with the grid.

Figure 2.9: Roombots (MOECKEL et al., 2013; BONARDI et al., 2013; VESPIGNANI et al., 2013; SPRöWITZ et al., 2014).

Each Roombot module can autonomously connect and disconnect from another module or dock to a passive connector embedded in the environment (BONARDI et al., 2013), but there are no external sensors for position and orientation in world-coordinates. Therefore, a human operator is needed to remotely control the joint movements through relative positions between joints (SPRöWITZ et al., 2014). In the future, absolute encoders are planned to be employed to detect joint backlash, and infrared sensors are devised to be included to give the estimation of distance and alignment of Roombots.

2.2.3.5 Transmote

Transmote (QIAO et al., 2012b; QIAO et al., 2012a) (shown in Figure 2.10) has an aluminum body shell and a hybrid lattice and chain architecture with three DoF. Each module has a docking mechanism with a male or female connector and can implement an inchworm-like crawling locomotion gait by coordinating the two end joints.

A set of two IR Sensors helps adjusting the robot position to align the male connector to the female connector, measuring the distance between docking interfaces. According to the characterization curves of the IR sensors, the threshold distance for docking is set to 9cm in misalignment in pitch and roll is not consider to simplify analysis.

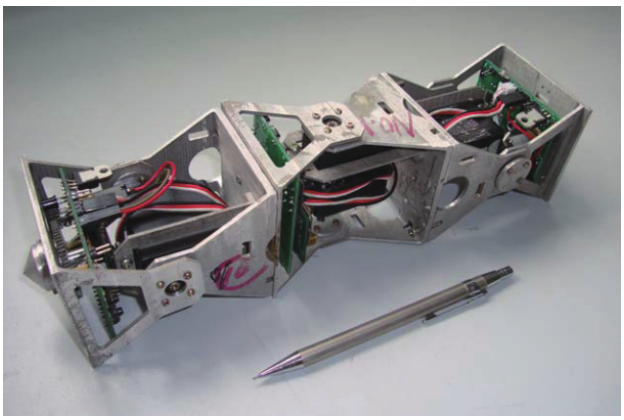


Figure 2.10: Transmote (QIAO et al., 2012b; QIAO et al., 2012a).

2.2.4 ErekoBot

Ereko Group is a research group of University of Brasilia with researchers from Mechanical and Computer Science departments. The group focus on the development of homogeneous modules with manual reconfiguration for inspection since 2009. The latest published prototype was the acrylic-shell *ErekoBot* $\alpha\, v.2$ module (SOUSA; VIANA; KOIKE, 2013a) with dimensions $50\,mm \times 50\,mm \times 50\,mm$ and weight 100 g (shown in Figure 2.11). For connection, each *ErekoBot* $\alpha\, v.2$ has Velcro^{TM} straps glued to the acrylic structure. This project aimed a small, light and cheap manually reconfigurable modular robot.

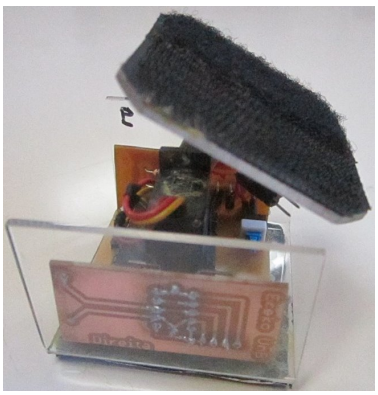


Figure 2.11: *ErekoBot* $\alpha\, v.2$.

In the embedded electronic circuit, there are two LiPo batteries to power the microcontroller and the servo motor. A PWM signal controls the servo motor position through a wired communication via USART.

2.3 General Measurement System in SRM Systems

We define a process as a system that generates information (BENTLEY, 2005), such as temperature, pressure, velocity, and displacement. The purpose of a measurement system is to link the observer (person who needs the information) to the process. We call the information *variable*,

where the input to the measurement system is the *true value*; and the output is the *measured value* of the variable. In an ideal measurement system, the measured value is equal to the true value, and the *accuracy* quantifies how close the measured value is to the true value.

In a given system, there are some types of element may be missing or may occur more than once (BENTLEY, 2005):

Sensing element This element gives an output that depends in some way on the measured variable.

Signal-conditioning element This element takes the output of the sensing element and coverts it into a form more suitable for further processing (DC voltage, DC current, frequency signal, for example).

Signal-processing element This element takes the output of the signal conditioning element and converts it into a form more suitable for presentation (for instance ADC, DAC).

Data representation element This element presents the measured value in a form recognizable for the observer.

As described in Section 2.2, SRM systems require on-board sensing elements to perform autonomous exploration of the environment. Therefore, we need to design each module measurement structure. In Section 2.3.1, we discuss characteristics that typical elements possess and their effect on the overall performance of the system. In addition, we describe some sensing elements tipically used in SRM robots for orientation (Section 2.3.2) and proximity (Section 2.3.3).

2.3.1 Systematic Characteristics

Mathematical or graphical tools can be extremely helpful to quantify the relationships which may occur between the output O and input I of an element when I is either at constant value or changing slowly(BENTLEY, 2005). Some characteristics are described as:

Range This characteristic is the minimum and maximum values of I and O (I_{MIN},I_{MAX},O_{MIN} and O_{MAX}).

Span This characteristic is the maximum variation in input or output ($I_{MAX} - I_{MIN}$ and $O_{MAX} - O_{MIN}$).

Ideal straight line An element is linear if the corresponding values of I and O lie on a straight line. The ideal straight line connects the minimum point $A(I_{MIN}, O_{MIN})$ to the maximum point $B(I_{MAX}, O_{MAX})$. In common usage, linearity refers to a mathematical relationship or function that can be graphically represented as a straight line, as in two quantities that are directly proportional to each other, such as voltage and current in a simple DC circuit, or the mass and weight of an object. When the relationship differs from that, the system is said to be non-linear.

Sensitivity This is the change ΔO in output O for unit change ΔI in input I, i.e. it is the ratio $\frac{\Delta O}{\Delta I}$.

Environmental effects In general, the output also depends on environmental inputs, such as temperature, pressure, humidity, supply voltage. There are two main types of environmental inputs: a modifying input and an interfering input. A modifying input causes the linear sensitivity of an element to change and an interfering input causes the straight line intercept or zero bias to change.

Hysteresis The output may change depending on whether the input is increasing or decreasing. Hysteresis is the dependence of the output of a system not only on its current input, but also on its history of past inputs. The dependence arises because the history affects the value of an internal state.

Resolution Some elements are characterized by the output increasing in a series of discrete steps in response to a continuous increase in the input. Resolution is the largest change in the input that can occur without any corresponding change in the output.

Calibration An element is calibrated by measuring and comparing values of the input I, the output O, the environmental input I_M and the environmental output O_M, when I is either at a constant value or changing slowly (BENTLEY, 2005). The accuracy of a measurement of a variable is the closeness of the measurement to the true value of the variable. We can quantify it by terms of measurement error (the difference between the measured value and the true value).

Accuracy and Precision Accuracy is a measure of how close to the actual value a measurement is. Precision is a measure of how close each measurement is to each other.

Lookup table A lookup table (LUT) is an array that replaces runtime computation with a simpler array indexing operation.

2.3.2 Rotation Motion sensors

Rotation sensors measure angular motion of a body about some rotation axis, providing attitude and orientation information commonly used in military, aerospace, industrial, medical and consumer areas (MORRIS, 2001). The main factors for choosing a sensing element for a specific application includes the analysis of sensibility, size, cost and resolution.

There are various devices available for measuring rotation, but the most used in SRM robots are accelerometers (Section 2.3.2.1) and gyroscopes (Section 2.3.2.2).

2.3.2.1 Accelerometer

Acceleration is an inert physical characteristic of any system, and control systems often use these measurements (in m/s^2) to correct dynamic conditions. An accelerometer is a device able to measure linear acceleration, local gravitational field, tilt and shocks as well as vibration.

Generally, an accelerometer consists of a proof mass suspended by compliant beams anchored to a fixed frame. The proof mass has a mass of m, the suspension beams have an effective spring constant stiffness k and there is a damping factor b affecting the dynamic movement of the mass generated by the air-structure interaction.

A second-order Mass-Damper-Spring system, as shown in Figure 2.12, models the accelerometer: an external acceleration displaces the support frame, which in turn changes the internal stress in the suspension spring. When an external force $F_{external}$ acts at the accelerometer, the proof mass develops a force given by D'Alembert inertial force $F_{inertial} = ma(t)$ (MORRIS, 2001), where m is the mass of the system and $a(t)$ the acceleration. This force also displays the spring by a distance $x(t)$, generating a damping $F_{damping}$ and a spring force F_{spring}. Hence the total force externally balance is the sum of internal forces:

$$F_{external} = F_{inertial} + F_{damping} + F_{spring}.$$

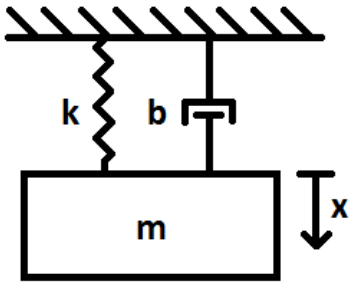


Figure 2.12: Accelerometer mass-spring-damper model.

The following second-order differential equation gives the dynamic equation of the vibration system (MORRIS, 2001):

$$m\,a(t) = m\frac{d^2}{dt^2}x(t) + b\frac{d}{dt}x(t) + kx(t).$$

Using Newton's second law and the accelerometer model, we obtain the mechanical transfer function

$$H(s) = \frac{X(s)}{A(s)} = \frac{m}{ms^2 + bs + k} = \frac{1}{s^2 + 2\zeta\omega_r s + \omega_r^2}, \tag{2.1}$$

where $\omega_r = \sqrt{\frac{k}{m}}$ is the natural frequency and $\zeta = \frac{b}{2\sqrt{km}}$ is the damping ratio.

Although Equation (2.1) rules its operation, there are still various types of accelerometers, based on materials and principles of operation, as shown below.

Capacitive There are two plates in the capacitor with sense electrodes, only one of them is fixed. When acceleration acts, it moves the non-moving plate and consequently causes a change in the capacitance between them and the sensor measures acceleration differences accordingly. Capacitive sensors have important advantages compared to other types of inertial sensors. They have simple structure and hence low fabrication cost. In addition, they provide low power consumption, high sensitivity, and high reliability as well as low nonlinearity, low

temperature dependency, low noise, and low drift (ERİŞMİŞ, 2004).

Hall Effect Changes in the magnetic field result in voltage variations, causing the Hall Effect (MORRIS, 2001). With a proof mass integrated at a semiconductor material, it is possible to measure the output of the hall element, which varies according to the applied force because of the consequent variation in the sensed magnetic field.

Piezoelectric The Piezoelectric Effect is the increase of electricity (or electric) polarity by applying a mechanical stress to certain crystals, because their asymmetrical lattice of molecules distorts when a force is applied (MORRIS, 2001). With a proof mass attached to a crystal, it is possible to measure the output voltage differences due to the acceleration.

Piezoresistive When strained, some metals changes its own electrical resistivity, which is called the Piezoresistive Effect (MORRIS, 2001). With a proof mass attached to this type of material, it is possible to measure the resistance difference generated due to the acceleration.

In 1989, Prof. R. Howe from Stanford University described a special type of technology, the Micro Eletromechanical Systems technology (MEMS), which consists of devices with features smaller than $100\,\mu m$. The most notable elements from this technology were the micro sensors, which not only were small, but also demonstrated performances exceeding the macro scale counterparts.

The most common MEMS accelerometers are capacitive because of its high sensitivity and resistance through temperature (LEE et al., 2005). As the micro scale type, they contain a movable proof mass with plates (capacitors) attached through a mechanical suspension system to a reference frame. With the changes at the capacitance, it is possible to know the deflection and acceleration of the proof mass.

2.3.2.2 Gyroscope

Gyroscopes measure both absolute angular displacement and absolute angular velocity, based on the principle of conservation of angular momentum. As can be seen in Figure 2.13, a gyroscope is essentially a spinning disk that rotates about the x, y and z axes (MORRIS, 2001), and, once the device spins, it tends to resist changes to its orientation due to the angular momentum of the wheel.

The unsupported end of the rotating gyroscope does not fall, but it starts to move in circular path about the vertical axis. There are two forces in this system, a normal force n, acting upwards at the pivot, and a downward gravitational force mg (JENSEN, 2011). The gravitational force produces a torque $\tau = mgh$, which relates, by definition, to the angular momentum M as

$$\tau = \frac{dM}{dt}. \tag{2.2}$$

Equation (2.2) states that the torque produces a change in angular momentum dM, which are in the same direction as τ and perpendicular to M. In summary, the gyroscope spins about one

axis, the gravitational force creates a torque about a second axis, and, as a result, the gyroscope rotates about a third axis. We call this motion the Precession Motion, and its effect the Gyroscope Effect.

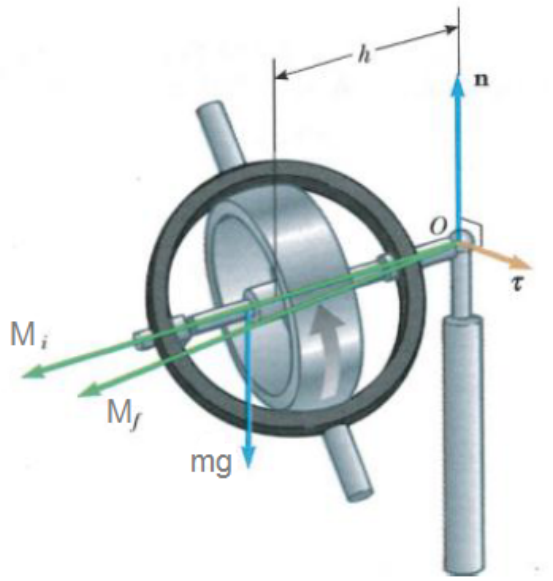


Figure 2.13: Gyroscope (Figure from (JENSEN, 2011)). n is a normal force acting upwards at the pivot O; mg is a downward gravitational force that produces a torque τ related to the angular momentum M.

In a time interval dt, the axis rotates an angle d, as the magnitude of $M_i \approx M_f \approx M$ remains the same, angular velocity can be calculated from the vector diagram shown in Figure 2.14

$$\sin(d\theta) \approx d\theta = \frac{dM}{M} = \frac{\tau dt}{M} = \frac{(mgh)dt}{M}.$$

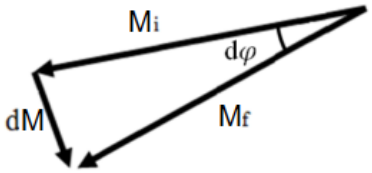


Figure 2.14: Vector diagram for a gyroscope.

Assuming small rotations $\sin(d\theta) \approx d\theta$ and dividing by $dt \to 0$, we describe the relation between torque $\tau = mgh$, angular momentum M and procession motion velocity ω_p as

$$\omega_p = \frac{d\theta}{dM} = \frac{mgh}{M}. \tag{2.3}$$

Equation (2.3) is useful to analyze the structural systems affected by gyroscope effects.

There are mainly two kinds of gyroscope: mechanical, depicted in Figure 2.15 and optical, illustrated in Figure 2.16.

Mechanical It consists essentially of a rotor driven wheel whose angular momentum is high enough to maintain the axis of rotation fixed in space, thus acting as a reference point (inner ring) (MORRIS, 2001). The inner ring is attached to the body whose motion it is desired to measure (frame). The output is the angle between the frame and the outer ring.

Optical It works with light beams instead of wheels, which results in high performance although it is rather difficult to fabricate. The major advantages of optical gyroscopes are that they are immune to electromagnetic interference and they can operate at high temperatures (ERİŞMİŞ, 2004). There are many types of optical gyroscopes, such as the ring laser, shown in Figure 2.16a, and the fibre-optic, presented in Figure 2.16b, both of them measuring the time and position of the beams of light and tubes. The ring laser gyroscope consists of laser gain tubes, mirror glasses and anode/cathodes that generate the laser beams. Any rotation of the systems changes the coherence of the beams, raising one and lowering the other. In the fibre-optic (Figure 2.16b), there are incident light from a source separated by a beam splitter into a pair of beams a and b. These beams travel in opposite directions around an optic-fibre coil and emerge from the coil marked as a' and b': any motion of the coil causes a phase shift between a' and b' detected by the interferometer (ERİŞMİŞ, 2004).

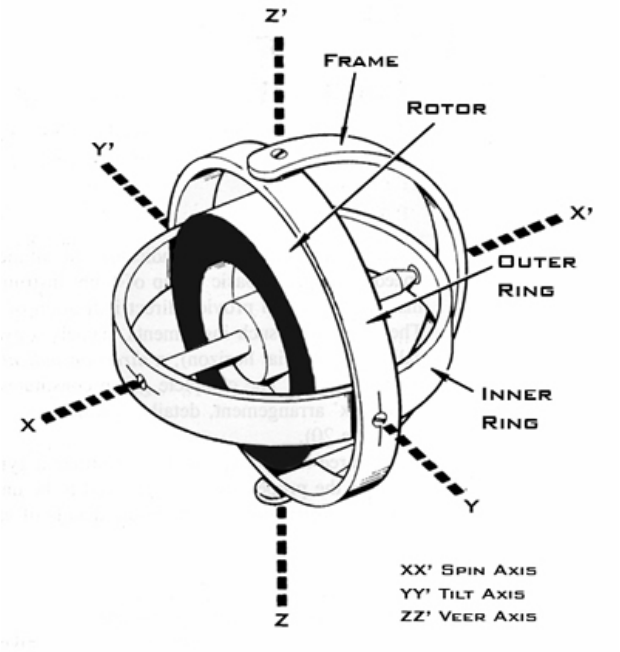
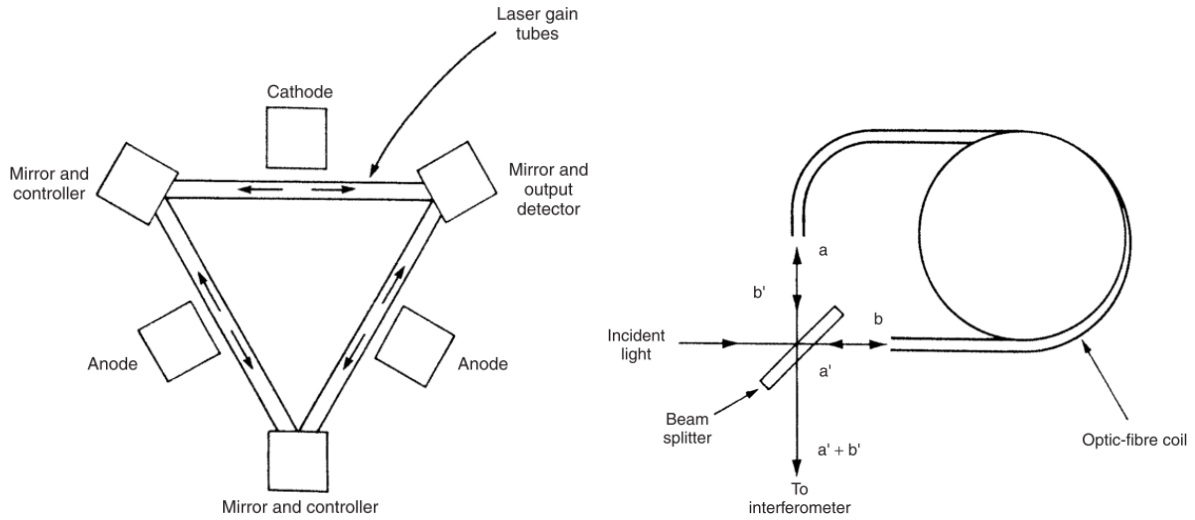


Figure 2.15: Mechanical Gyroscope.

In the last decade, industry opted for the MEMS gyroscopes, which are extremely small and accurate for applications of orientation and tilting. This technology, with vertically driven vibrating masses, produces a functionally complete, low-cost motion sensor.



(a) Ring Laser Gyroscope. (b) Fibre-Optic Gyroscope.

Figure 2.16: Optical Gyroscope (Figures from (MORRIS, 2001)).

2.3.3 Translational Displacement Sensors

Translational displacement sensors detect the motion of a body in a straight line between two points using electromagnetic fields, light, and/or sound without any physical contact (MORRIS, 2001). Usually, a translational displacement sensor emits an electromagnetic field or a beam of electromagnetic radiation and search for changes in the field or in the returned signal.

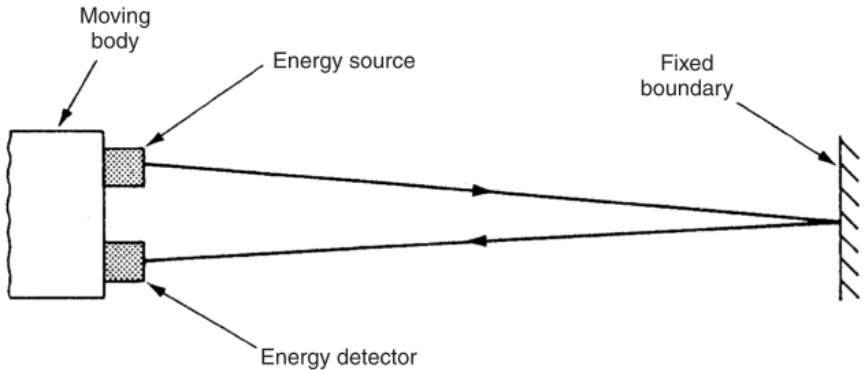
As in the case of orientation sensors, it is necessary to analyze sensibility, size, cost and resolution when choosing a translational displacement sensor for specific applications and environments.

Some techniques for measuring large translational displacements exist and measure motions of a body with respect to some fixed initial point. *Range sensors* provide a well-used technique of measuring the translational displacement of a body with respect to some fixed boundary. Figure 2.17 illustrates how range sensors work.

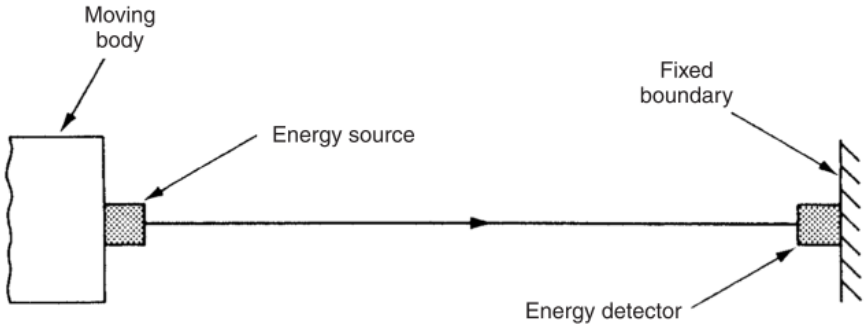
Range sensing systems consists of an energy source, an energy detector and electronic means of timing the time of flight of the energy between the source and detector. The form of energy used is either ultrasound or light. In some systems, both energy source and detector are fixed on the moving body and operation depends on the energy being reflected back from the fixed boundary as in Figure 2.17a. In other systems, the energy source is attached to the moving body and the energy detector within the fixed boundary, as shown in Figure 2.17b.

2.3.4 Filters

Signal filtering consists of processing a signal to remove a certain band of frequencies within it (MORRIS, 2001). The band removed can be either at the low-frequency band (low-pass filter, Figure 2.18a) or at the high-frequency band (high-pass filter, Figure 2.18b), at both bands (band-pass filters, Figure 2.18c), or in the middle (band-stop filters, Figure 2.18d). Either analogue or



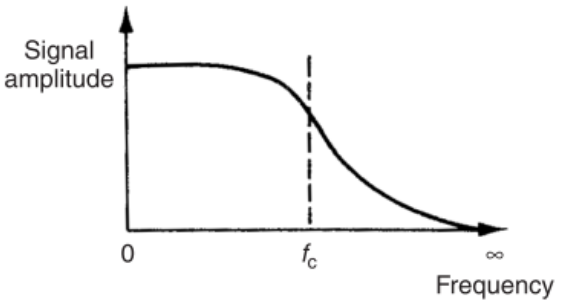
(a) Energy source and detector fixed on the moving body.



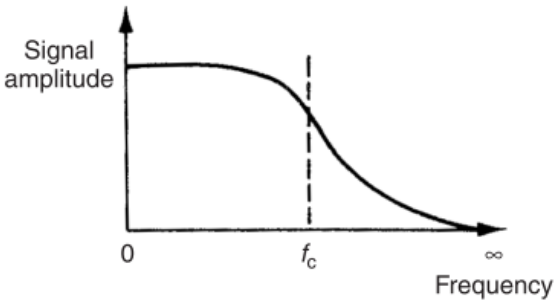
(b) Energy source attached to the moving body and detector to the fixed boundary.

Figure 2.17: Range Sensors (Figures from (MORRIS, 2001)).

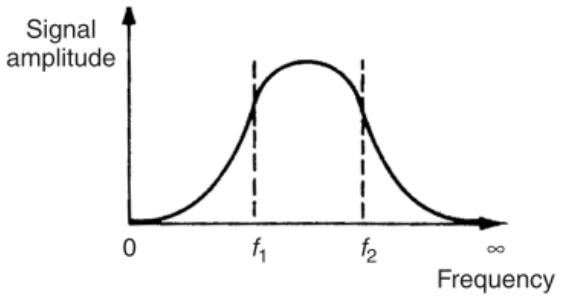
digital methods can filter signals.



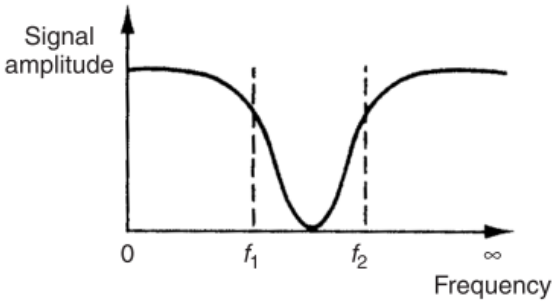
(a) Low-pass filter.



(b) High-pass filter.



(c) Band-pass filter.



(d) Band-stop filter.

Figure 2.18: Outputs from filters (Figures from (MORRIS, 2001)).

An analogue filter is an electrical network, consisting usually of resistors, capacitors and operational amplifiers, which conditions continuous signals; and a digital filter is usually a digital computer programmed to process sampled values of a signal (BENTLEY, 2005).

Since single results of sensors lack of accuracy, there are filters that combines the outputs of gyroscope and accelerometer. The Complementary Filter is an estimation technique widely used in robotics to combine measurements (Section 2.3.4.1). This filter is actually a Kalman Filter at steady state (Wiener filter) (KOVVALI; BANAVAR; SPANIAS, 2013) for a class of filtering problems (HIGGINS, 1975). Users of the Complementary Filter disregard statistical descriptions of the noise and obtain its filter by a simple analysis in the frequency domain. On the other hand, users of the Kalman Filter work in the time domain and disregard the transfer function or frequency domain approach to the filtering problem.

2.3.4.1 Complementary Filter

Usually, we call Complementary Filter any digital algorithm used to blend or merge similar or redundant data from different sensors to achieve a robust estimate of a single state variable. In addition, while filters usually act on the signal, the complementary filter acts only on the different kinds of noise associated with different kinds of measurements of the same signal.

Assume a typical system with two inputs, where one input gives information with a high frequency noise, and then goes through a low frequency filter. In addition, a second input gives

information with a low frequency noise, and then goes through a high frequency filter. If the filters are mathematically complementary, the filter output is a complete reconstruction from the measurement variable without the noise (GLASSER,). Figure 2.19 illustrates this process with a perfect low pass filter and a perfect high pass filter.

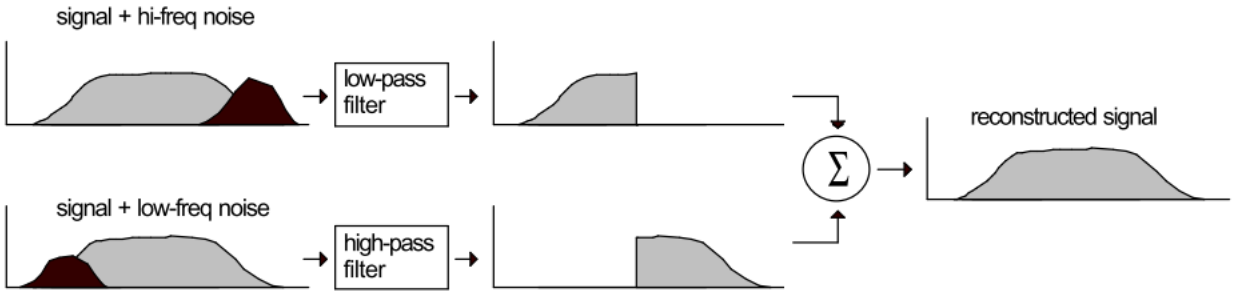


Figure 2.19: Signals from a complementary filter (Figure from (GLASSER,)).

As the Complementary Filter is a filter in the frequency domain, it uses two or more transfer functions that are mathematically complementary to one another. Then, if $G(s)$ is associated to a sensor, and $I - G(s)$ is associated to another sensor, then the sum of the transfer functions is I (the identity matrix).

It can be seen in Figure 2.20 a basic complementary filter, where x and y are measurements from z with noise, and $\hat{z}$ is the estimate value from z.

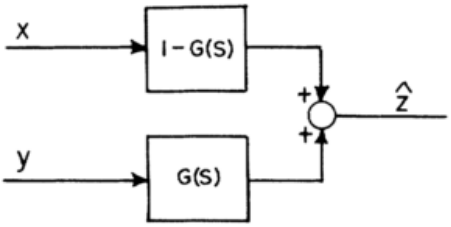


Figure 2.20: A basic complementary filter (Figure from (HIGGINS, 1975)).

Considering that n_1 is the noise that acts on signal x and n_2 on y, the filter can also be set as shown in Figure 2.21. In this case, the input for $G(s)$ is $y - x = n_2 - n_1$, therefore, $G(s)$ only operates at the noise or error from x and y.

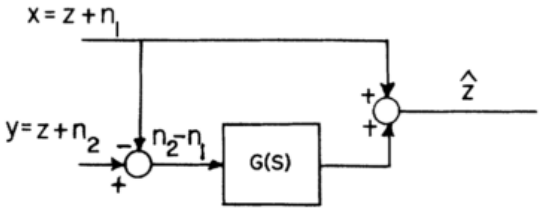


Figure 2.21: Alternative complementary filter block diagram (Figure from (HIGGINS, 1975)).

A typical example of application is the fusion between vertical acceleration and barometric vertical velocity measurements to obtain an estimate of vertical velocity. The acceleration measurement $\ddot{h}_a$ is integrated producing a signal $\dot{h}_a$, and this integration attenuates the high-frequency

noise, whereas the noise in $\dot{h}_b$ remains the same. Figure 2.22 illustrates the process of filtering $\dot{h}_b$ with the low-pass filter

$$G(s) = \frac{1}{\tau s + 1}$$

and $\dot{h}_a$ with the high-pass filter

$$1 - G(s) = 1 - \frac{1}{\tau s + 1} = \frac{\tau s}{\tau s + 1}.$$

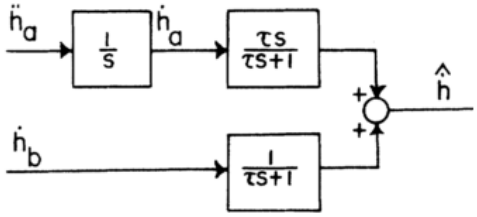


Figure 2.22: Complementary filter estimating vertical velocity (Figure from (HIGGINS, 1975)).

2.4 Orientation: Euler Angles

2.4.1 Euler angles

Euler angles describe rotation as a sequence of rotations about three mutually orthogonal coordinate axes fixed in space. These axes may be world or local coordinates, where rotations act on points in the space (USTA, 1999). The coordinate axes remain fixed, and subsequent rotations have the effect of rotating the axes about the preceding rotations. There are six possible ways to order three sequential rotations on different axes, different rotations lead to different orientations.

Euler angles method defines the rotation on three body fixed orthogonal axes by extending 2D rotations to 3D for each axis. Figure 2.23 shows Euler angles rotations and Equations (2.4), (2.5) and (2.6) represent individual rotation matrices.

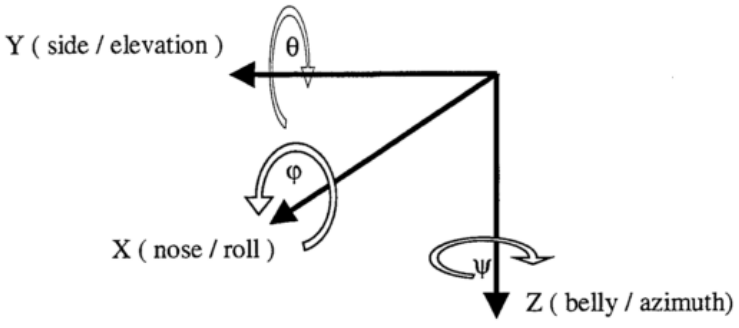


Figure 2.23: Rotation in 2D (Figure from (USTA, 1999)).

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{2.4}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.5}$$

$$R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

Matrix multiplication of these matrices produces rotation matrices for multiple rotations. It is important to notice that different sequences result in different rotation matrices.

2.4.1.1 Gimbal Lock

Gimbal lock is the loss of one degree of freedom in a three-dimensional, three-gimbal mechanism that occurs when the axes of two of the three gimbals are driven into a parallel configuration, locking the system into rotation in a degenerate two-dimensional space. There are two common situations of Gimbal lock:

- When is not possible to represent the orientation of the sensor using Euler Angles. The exact orientation at which gimbal lock occurs depends on the order of rotations used (Equations (2.4), (2.5) and (2.6)). It may occur that, after some operations, two axes may stay align, *locking* the system into rotation in a degenerate two-dimensional space, losing one degree of freedom. One way to avoid the Gimbal Lock is to use quaternions for representing rotations in 3D space.
- In gyroscopes, gimbal lock may occur when vehicle rotation causes two of the three gimbal rings to align with their pivot axes in a single plane. When this occurs, it is no longer possible to maintain the sensing platform's orientation.

2.5 Summary

This chapter has described pipeline inspection and the latest pipeline robots. Also, it was presented the problem of flexibility and, in contrast to the classical solutions using rigid structures, the idea has arisen of using SRM robots, which are capable, at any moment, of changing their shape to be able to move in the most efficient way.

To sum up, the chapter mentioned important definitions and concepts of general measurement system, using systematic characteristics to describe some rotation motion and translational

displacement sensors. The specific sensors in which this work is interested are accelerometer, gyroscope and IR sensors. These sensors need filters for processing their information in order to allow some robot tasks. In order to calculate the orientation pose of the modular robot, definitions and concepts of the orientation technique using Euler angles were also presented.

Although other investigators have constructed prototypes of SRM robots, up to now there are few works with the construction of a low-cost module with cheap sensors and simple algorithms for alignment with an external plane.

Chapter 3

Conceptual Design and Modelling

3.1 Introduction

After the research background presented in Chapter 2, this chapter deals with concepts and models employed for the study of sensing in SRM robotics. Section 3.2 presents the concept of the *ErekoBot* σ introduced and, then, Section 3.3 demonstrates the fundamental measurement system models: attitude, distance and filtering. Section 3.4 proposes the Alignment Algorithm using the response from IR sensors and IMU. Finally, the conclusion will establish the relationship between all of these and propose a solution to the sensory problem in SRM robotics.

3.2 Conceptual Design of *ErekoBot* σ

The conception, design and manufacturing of a robotic platform specific for a task consumes a lot of time and resources, and the resulting robot is obviously very expensive. Therefore, there is a comprehensible concern about its integrity: it is usual to make it robust to support hazard environments, which makes them even more expensive. In order to reduce the impact of these concerns, the *ErekoBot* σ was conceived to be a simple and robust Reconfigurable Modular Robot with embedded sensors suitable for pipeline inspection. The important contribution of this work regarding SRM robotics is the addition of an inexpensive but efficient measurement system, making it less expensive than the robots presented in Sections 2.1 and 2.2. This work focus on sensing improvements of the *ErekoBot* $\alpha\, v.2$, and therefore, the robot is still manually reconfigurable (for total self-reconfigurability the type of connection, locomotion and algorithms of reconfigurability need to be upgraded).

For simplicity, the module is represented as shown in Figure 3.1: two segments (or linkages) with total length L and mass m. A joint links both segments, determining a bending angle (φ), which is restricted in the interval $[-90, 90]$ degrees (typical of commercial servos).

Consider a snake robot, composed of modules manually configured by a user, and inspecting a pipeline, as shown in Figure 3.2. When *ErekoBot* σ system finds a possible point of failure, with

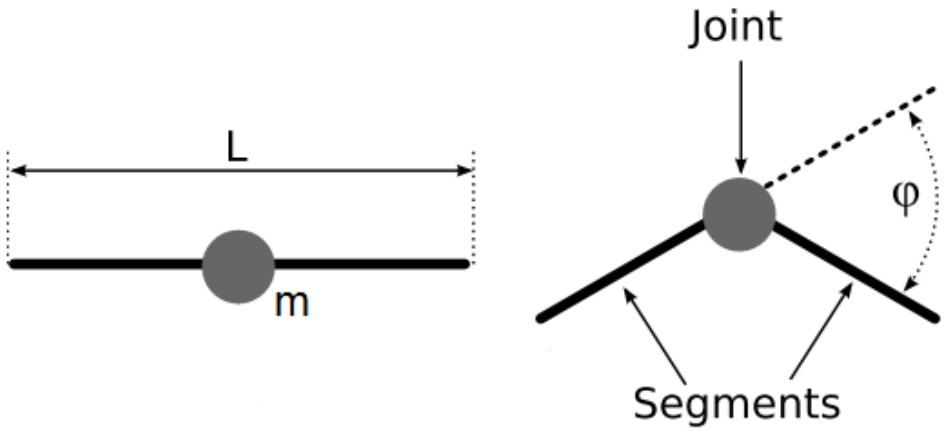


Figure 3.1: Wired model of the module.

a camera for example, the robot needs to align itself with the pipeline internal walls and make a complete inspection of the area.

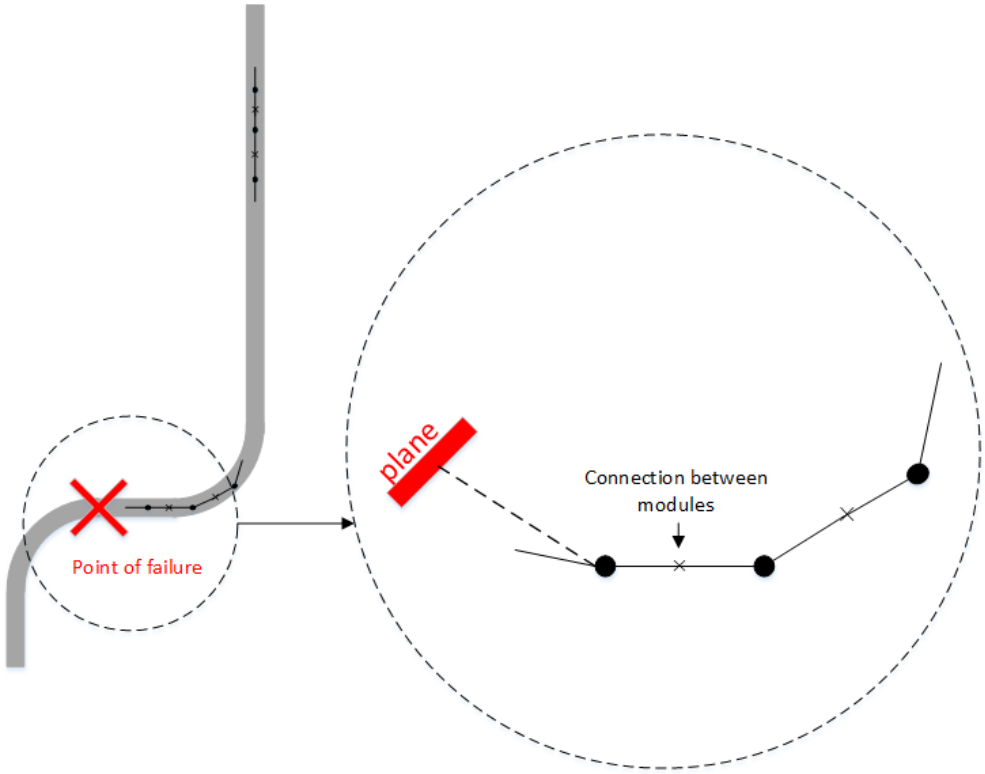


Figure 3.2: *ErekoBot* σ example of application.

Each *ErekoBot* σ must have the ability to (1) detect an obstacle (Figure 3.3a), (2) align with a plane simulating a pipeline (Figure 3.3b) and (3) estimate its own pose (Figure 3.3c).

The basic parameters whose requirements were necessary for the module design are shown in Table 3.1. These requirements are in accordance with *ErekoBot* $\alpha\, v.2$ described in Section 2.2.4

The 3D-model was designed by Ricardo Diniz Caldas, including microcontroller, servomotor

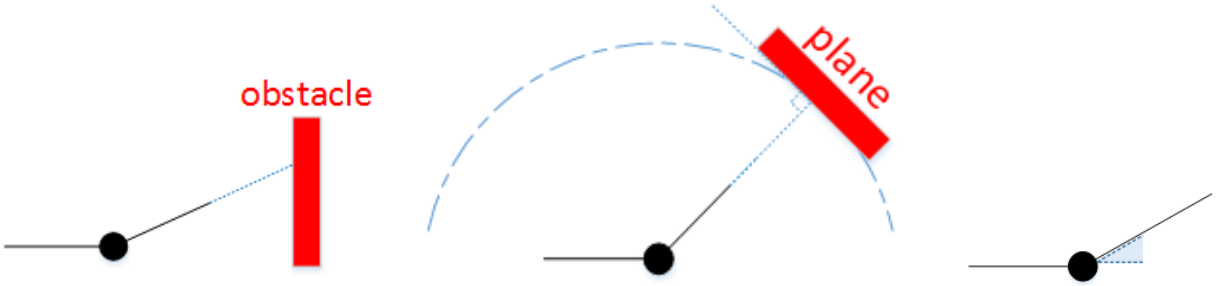


Figure 3.3: *ErekoBot* σ requirements.

Parameter	Requirement
Dimensions	bigger than $50\,mm \times 50\,mm \times 50\,mm$
Microcontroller	AVR ATMega series
Number of DoF	1 (rotational 180°)
Motor	Servo motor

Table 3.1: Basic requirements for *ErekoBot* σ design.

and sensors (Figure 3.4).

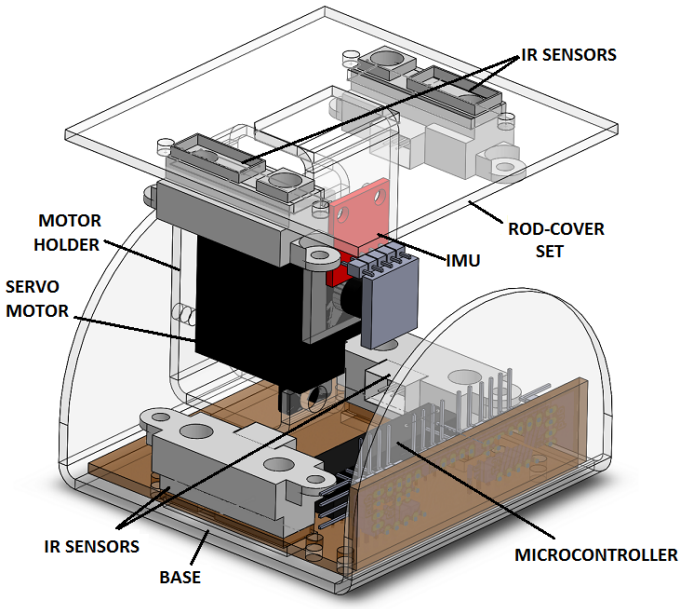


Figure 3.4: *ErekoBot* σ 3D model.

3.3 Instrumentation Model

Section 3.3.1 presents a method to calculate the attitude of a body with few calculations, and it is proposed here that these calculations be based on gyroscope and accelerometer response from an Inertial Measurement Unit (IMU). In addition, a filter is proposed to combine the angles resulting

from the gyroscope and accelerometer (Section 3.3.1.3). Finally, Section 3.3.2, proposes a fusion processing technique of data measured through the IR sensors within a limited range.

3.3.1 Attitude Estimation

The Euler angles are calculated from the gyroscope and accelerometer readings based on equations of the Rotation Matrices, presented at Section 2.4.1. The attitude estimator is a filter that combines the different angles (from the gyroscope and accelerometer) into one filtered angle (Section 3.3.1.3), and is estimating the roll and pitch angles.

The *ErekoBot* σ reference frame is the body frame denoted with a subscript b. The navigation frame is a fixed frame denoted with a subscript n, as showed in Figure 3.5.

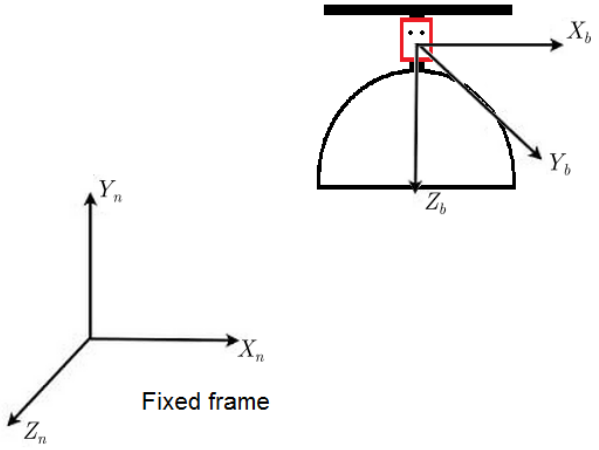


Figure 3.5: Frames with positive axes and torques.

A rigid body can be placed in an arbitrary orientation by first rotating it about its z-axis by an angle ϕ (azimuth or yaw rotation), then about its y-axis by an angle θ (elevation or pitch rotation), and finally about its x-axis by angle ϕ (bank or roll rotation).

3.3.1.1 Gyroscope Model

The ideal gyroscope gives the raw measurements values $\vec{g}_b = [g_x, g_y, g_z]^T$ in LSB (Least Significant Bit unit), this data ($Gyro_{scale}$) is scaled to convert to angular velocities in rad/s (MAGNUSSEN; OTTESTAD; HOVLAND, 2013):

$$\vec{\omega}_b = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} Gyro_{scale}.$$

Making a numerical integration of the gyroscope measurements results in

$$\phi_{gyro_t} = \phi_{gyro_{t-1}} + \omega_x \Delta t, \tag{3.1}$$

$$\theta_{gyro_t} = \theta_{gyro_{t-1}} + \omega_y \Delta t \tag{3.2}$$

and

$$\psi_{gyro_t} = \psi_{gyro_{t-1}} + \omega_z \Delta t, \tag{3.3}$$

where ϕ_{gyro_t}, θ_{gyro_t} and ψ_{gyro_t} are angles in the iteration t; $\phi_{gyro_{t-1}}$, $\theta_{gyro_{t-1}}$ and $\psi_{gyro_{t-1}}$ are angle in the iteration $t-1$; ω_x, ω_y and ω_z are angular velocities measurements in x, y and z by the gyroscope; and Δt is the sample time between each iteration.

The initial values ϕ_{gyro_0}, θ_{gyro_0} and ψ_{gyro_0} have to be set to arbitrary normalized angles, i.e. $[0^\circ\ 0^\circ\ 0^\circ]^T$, or to initial angles values from the accelerometer - which results in more correct initial values.

3.3.1.2 Accelerometer Model

The accelerometer values describes the body orientation vector relative to the gravity in the navigation frame, given a steady state of the movement. Sudden movements may affect the accelerometer with other forces and give a false reading of the gravity vector.

The measured acceleration vector in the body coordinate system is

$$\vec{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}.$$

A three-axis accelerometer mounted in a body oriented in the earth gravitational field will have output G_p given by

$$\vec{G}_p = \begin{bmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{bmatrix} = \vec{R}(\vec{g} - \vec{a}) = \vec{R}\vec{g} = \vec{R} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \tag{3.4}$$

where $\vec{R}$ is the rotation matrix describing the orientation of the module relative to the earth coordinate frame and $\vec{g}$ the gravitational field. It is assumed that the module has no linear acceleration (or it is small enough to be approximated to zero) and that its initial orientation is aligned with the z axis of the gravitational field, therefore $\vec{a} = [000]^T$. To define the roll, pitch and yaw rotations from an initial position, the matrices defined in Equations (2.4), (2.5) and (2.6) in Section 2.4.1 are employed. We also consider $\vec{g} = [001]^T$, using units of g.

Therefore, there are six possible rotations (as demonstrated in Appendix I.1). A direct con-

sequence for these differences is that roll, pitch and yaw rotation angles are meaningless without first defining the order in which we applied the rotations (PEDLEY, 2013).

Four sequences can be immediately rejected: the system has only two degrees of freedom since the vector magnitude must always equal 1 in the absence of a linear acceleration. It is not possible to solve for three unique values of roll, pitch and yaw angles.

All accelerometers are completely insensitive to rotations about the gravitational field vector, and it cannot be used to determine the yaw rotation. Therefore, it is conventional to select either the rotation sequence R_{xyz} (Equation (3.5)) or R_{yxz} (Equation (3.6)) to eliminate the yaw rotation (PEDLEY, 2013).

$$\vec{R}_{xyz}\begin{bmatrix}0\\0\\1\end{bmatrix} = \vec{R}_x(\phi)\vec{R}_y(\theta)\vec{R}_z(\psi)\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}-\sin\theta\\ \cos\theta\sin\phi\\ \cos\theta\cos\phi\end{bmatrix} \tag{3.5}$$

$$\vec{R}_{yxz}\begin{bmatrix}0\\0\\1\end{bmatrix} = \vec{R}_y(\theta)\vec{R}_x(\phi)\vec{R}_z(\psi)\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}-\sin\theta\cos\phi\\ \sin\phi\\ \cos\theta\cos\phi\end{bmatrix} \tag{3.6}$$

The Equation (3.5) can be rewritten in the form

$$\frac{\vec{G_p}}{||G_p||} = \begin{bmatrix}-\sin\theta\\ \cos\theta\sin\phi\\ \cos\theta\cos\phi\end{bmatrix} \Rightarrow \frac{1}{\sqrt{G_{px}^2+G_{py}^2+G_{pz}^2}}\begin{bmatrix}G_{px}\\G_{py}\\G_{pz}\end{bmatrix} = \begin{bmatrix}-\sin\theta\\ \cos\theta\sin\phi\\ \cos\theta\cos\phi\end{bmatrix}, \tag{3.7}$$

relating the roll ϕ and pitch θ angles to the normalized accelerometer reading G_p. Solving for roll and pitch angles, it is obtained

$$\tan(\phi_{xyz}) = \frac{G_{py}}{G_{pz}} \tag{3.8}$$

$$\tan(\theta_{xyz}) = \frac{-G_{px}}{G_{py}\sin\phi + G_{pz}\cos\phi} = \frac{-G_{px}}{\sqrt{G_{py}^2+G_{pz}^2}} \tag{3.9}$$

The Equation (3.6) can be rewritten in the form

$$\frac{\vec{G_p}}{||G_p||} = \begin{bmatrix}-\sin\theta\cos\phi\\ \sin\phi\\ \cos\theta\cos\phi\end{bmatrix} \Rightarrow \frac{1}{\sqrt{G_{px}^2+G_{py}^2+G_{pz}^2}}\begin{bmatrix}G_{px}\\G_{py}\\G_{pz}\end{bmatrix} = \begin{bmatrix}-\sin\theta\cos\phi\\ \sin\phi\\ \cos\theta\cos\phi\end{bmatrix}, \tag{3.10}$$

relating the roll ϕ and pitch θ angles to the normalized accelerometer reading G_p. Solving for roll and pitch angles, it results

$$\tan(\phi_{yxz}) = \frac{G_{py}}{\sqrt{G^2px + G^2pz}}, \tag{3.11}$$

$$\tan(\theta_{yxz}) = \frac{-G_{px}}{G_{pz}}. \tag{3.12}$$

The subscripts xyz and yxz denote the specific rotation employed to compute the angles.

Equations (3.8) to (3.12) have an infinite number of solutions at multiples of 360°. The range of the roll angle will be restricted to −180° to 180° and the pitch angle to −90° to 90° for the R_{xyz} rotation.

For convenience, the accelerometer outputs are normalized to unit vectors: $\tilde{\vec{a}}$ denote the normalized vector of the acceleration measurements,

$$\tilde{\vec{a}} = \frac{\vec{a}}{|a|} = \begin{bmatrix} \tilde{a}_x \\ \tilde{a}_y \\ \tilde{a}_z \end{bmatrix},$$

where $|a|$ is the norm of the acceleration vector $\vec{a}$.

3.3.1.3 Filter Model

A complementary filter is proposed to combine two different angles, resulting from the gyroscope and accelerometer ($angle^g$ and $angle^a$), into one filtered result to estimate roll and pitch angles (MAGNUSSEN; OTTESTAD; HOVLAND, 2013).

The gyroscope provides an angle with fast and smooth updates. The result calculated is a rotation from its initial position based on angular velocities; therefore, it is independent from the fixed reference frame. However, $angle^g$ drifts over time if not compensated.

The accelerometer always has a fixed reference frame: the navigation frame. However, the accelerometer is sensitive to noise and airframe vibrations and is not able to estimate the attitude as smooth and fast as the gyroscope.

The proposed filter combines $angle^g$ with $angle^a$ and compensate for both drifting and noise. As $angle^g$ starts to drift, it will be significant different from $angle^a$. We have to rotate $angle^g$ back towards $angle^a$, however, since $angle^a$ has a lot of noise, $angle^g$ should only be rotated enough to compensate for the drifting. A simple way of doing it is to compare $angle^g$ and $angle^a$ by calculating the difference $angle^e$:

$$angle^e = angle^a - angle^g,$$

where $angle^e$ is the correction needed to be added to $angle^g$ in order to become $angle^a$. The amount to rotate depends on a scaling factor P_{err}, so that at each cycle the estimated $angle$ becomes:

$$
\begin{aligned}
angle &= angle^g + P_{err}\, angle^e \\
angle &= angle^g + P_{err}\ (angle^a - angle^g) \\
angle &= angle^g \cdot (1 - P_{err}) + angle^a \cdot P_{err}
\end{aligned} \tag{3.13}
$$

where it is necessary to set $P_{err} \in [0, 1]$ as low as possible to reduce the noise from the accelerometer, but high enough to correct the drifting of the gyroscope. Setting $P_{err} = 0$ eliminates the accelerometer and $P_{err} = 1$ eliminates the gyroscope.

The value of P_{err} depends on factors such as gyroscope bias, cycle time of the system, accelerometer noise, etc. It is possible to find the initial value of P_{err} by monitoring the estimated angles with no movement: P_{err} is reduced until the estimated angle starts to drift. Figure 3.6 shows the block diagram of the filter.

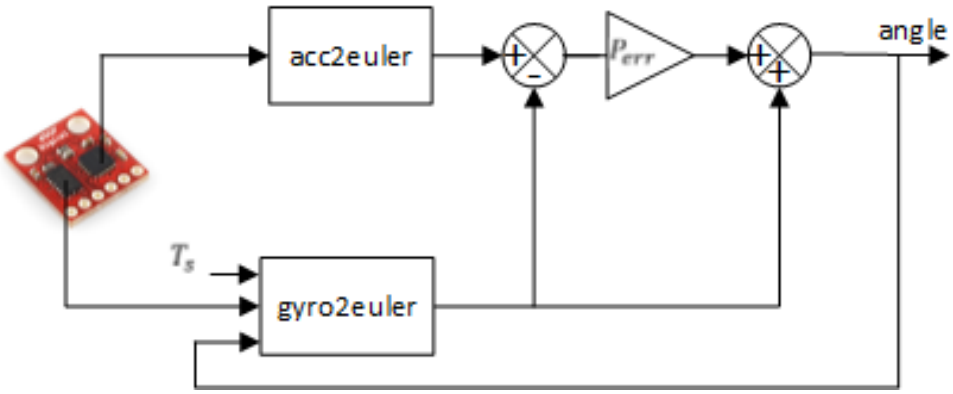


Figure 3.6: Block diagram of the attitude estimator (Figure adapted from (MAGNUSSEN; OTTESTAD; HOVLAND, 2013)).

The block *acc2euler* converts the accelerometer values from the accelerometer using Equations (3.8) and (3.9).

3.3.2 Distance Estimation

In order to position one face of the module parallel to a surface, this work proposes a method that adjusts the normal directions of the *ErekoBot* σ with respect to the surface of the object by detecting the error in the object attitude, using just IR sensors.

Consider the case where the surface of the object is larger than the surface of the *ErekoBot* σ , as shown in Figure 3.7. When the surface of the object is sufficiently larger than the surface of the sensor, and this relationship holds even if the sensor surface has a constant curvature, the attitude output is equivalent to position output, and it is feasible ou reasonable to correct to attitude error.

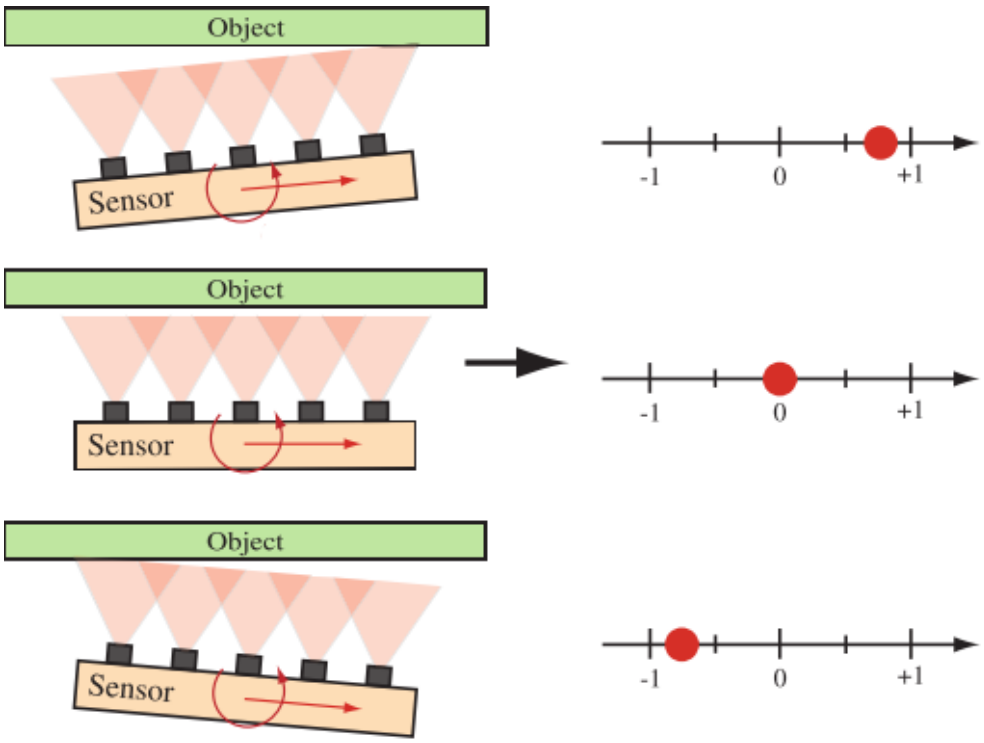


Figure 3.7: Sensor output depending on object attitude (Figure adapted from (YE et al., 2013)).

3.4 Alignment Algorithm

At each loop, the *ErekoBot* σ repeats the sequence:

1. Read the orientation of the rod at time t;
2. Read the objects distance at time t;
3. Send the orientation, distance and t to the host;
4. If the distance is different, move the servo motor to align, reducing the distance difference;
5. Add 1 to t and returns to step 1.

Above, the orientation is six degrees orientation readings from the gyroscope and accelerometer, the distance is the distances readings from the four IR sensors and t is the current time.

The host converts the IMU readings into angles and applies the complementary filter from Section 3.3.1.3. With the orientation angles and IR response, we can visualize at the screen the process of alignment.

3.5 Conclusions

Simplicity and robustness characterize the *ErekoBot* σ conceptual design, which avoids high costs, both in materials and processing. A module is represented as two segments lines, and these

lines connected in a 3D coordinate frame represent the complete reconfigurable modular robot.

Some basic parameters (such as dimensions, type of microcontroller and motor and number of degrees of freedom) were defined in order to start manufacturing a robot that follows its three main requirements.

When estimating the robot pose, it is proposed an attitude model using Euler angles with one gyroscope and one accelerometer. As gyroscopes and accelerometers have different and complementary advantages and disadvantages, a complementary filter is applied to combine these data for better response. For alignment, a method is proposed to change the *ErekoBot* σ position while the IR sensors are unaligned with the plane.

From these conceptual design and models, the *ErekoBot* σ was designed and, then, algorithms for alignment were implemented and tested (Chapter 4).

Chapter 4

Module Design

4.1 Introduction

As described in Chapter 1, the aim of this work is to develop a reconfigurable modular robot with sensors for pipeline inspections. This chapter shows the efforts made to construct the conceptual design of *ErekoBot* σ presented in Section 3.The selection of the sensor is described in depth in Section 4.2, as the electronic design is discussed in Section 4.3 and the mechanical design detailled in Section 4.4.

Figure 4.1 illustrates in detail the flowchart used in the phases of designing and manufacturing of *ErekoBot* σ : the choice of the sensors , the microcontroller and communication hardware, the intermodular connection, the electronic devices, the servo motor, the materials and the power supply are made with the help of a careful analysis if the final weight was acceptable. All the pieces of the module interfere at the final weight, but only the pieces with higher influence were analyzed. Finally, the pieces were manufactured and the parts assembled.

4.2 Sensors Selection

When choosing the sensors, the *ErekoBot* σ requirements, described in Section 3.2, were analyzed: (1) detect an obstacle (Figure 3.3a), (2) align with a plane (Figure 3.3b) and (3) estimate its own pose (Figure 3.3c).

Considering that the *ErekoBot* σ needs to detect obstacles at its front and its back, at least two translational displacement sensors are necessary, one at each side. However, the second requirement involves at least one more translational displacement sensor at each side for the distance estimator model described in Section 3.3.2. Finally, for the third requirement, *ErekoBot* σ needs rotation motion sensors attached to its mobile piece.

Therefore, 4 Infrared Sensors (IR Sensors) SHARP GP2Y0A41SK0F were choosen (Figure 4.2a). One of the strong points of these sensors is that the environmental temperature and operating duration do not influence this sensor unit. The voltage output is proportional to the detection

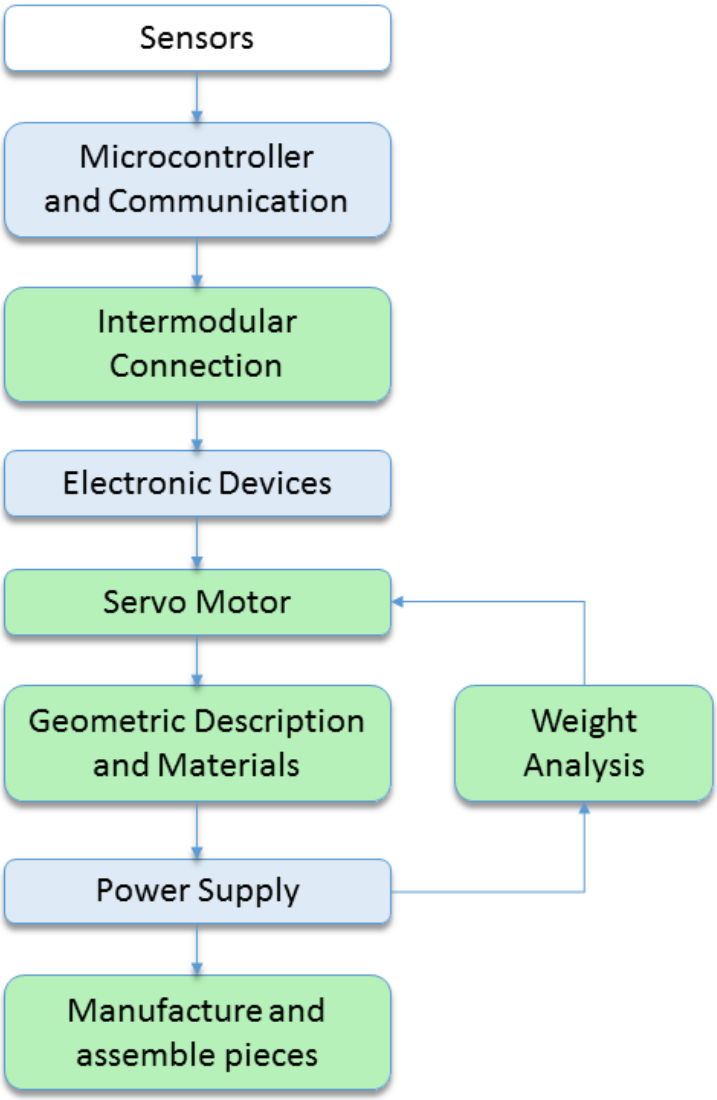
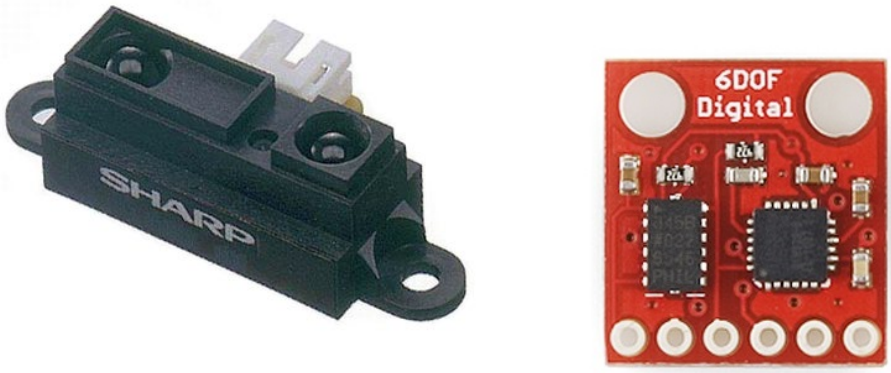


Figure 4.1: Flow chart for the *ErekoBot* σ design (Blue: Electronic Design; Green: Mechanical Design.).

distance (SHARP, 2002).

In addition, an inertial measurement unit (IMU) was selected, with one accelerometer ADXL345 (Analog Devices Inc, 2009) and one gyroscope ITG-3200 (INVENSENSE, 2010). This unit (Figure 4.2b is tiny, with two mounting holes and it communicates though Inter-Integrated Circuit (I2C) interface[1].

These sensors were elected mainly because of their repeatability, temperature stability, design flexibility, low cost, low power usage and, most of all, because of their small size.



(a) SHARP GP2Y0A41SK0F. (b) ADXL345 and ITG-3200.

Figure 4.2: *ErekoBot* σ sensors.

4.3 Eletronic Design

The electronic design consists in selecting the microcontroller and the communication architecture, as well as the possible types of power supply for each *ErekoBot* σ .

4.3.1 Microcontroller

Inside each module, the microcontroller controls communications, servomotor positions and sensors response. The criteria employed to decide upon a microcontroller are: number of pins, size of program memory, low-power consumption, low price and availability purchase in our country.

The *ErekoBot* $\alpha\, v.2$ uses an Atmel® AVR® ATmega8 , however the size of the program memory (8 Kbytes) was found to be insufficient for the sensors response, mainly because of the floating point number library. Therefore, another microcontroller from the same family was adopted, the Atmel® AVR® ATmega32 (8-bit RISC CPU, 32KB, 16MHz, USART communication and I2C interface), as illustrated in Figure 4.3a. The number and positions of pins are the same for both

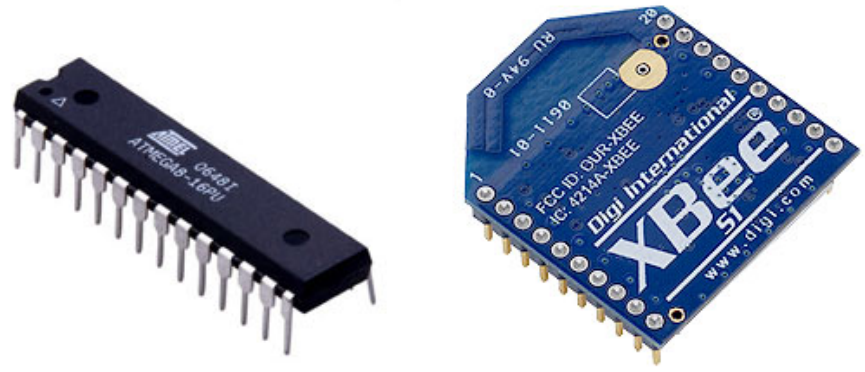
[1]I2C is a serial protocol for two-wire interface to connect low-speed devices like microcontrollers, EEPROMs, A/D and D/A converters, I/O interfaces and other similar peripherals in embedded systems.

microcontrollers, which simplifies the changes in the electronic design (ATMEL, 2011).

The Atmel® AVR® ATmega32 does not contain a floating-point unit (FPU), so additional libraries are required to compute it. However, these libraries occupy a relatively large space in code size and can compromise the execution speed. Nevertheless, the Atmel® AVR® ATmega32 worked in our tests due to its simplicity, low-power consumption, low-price and availability.

4.3.2 Communication

For a reliable and simple RF communication between the modules and the operator outside the pipeline, one Digi® XBee® 1mW PCB radio module, as shown in Figure 4.3b, is employed at each *ErekoBot* σ , as it provides two serial modes of communication: a simple transmit/receive method and a framed mode that offers advanced features (Digi International Inc, 2009). Even with the small PCB antenna, this XBee has an outdoor range up to $90\,m$ and an indoor range up to $30\,m$. One XBee is connected at each module and another one at the host, allowing transmission between all parts of the system through a serial port (Figure 4.4).



(a) Atmel® AVR® ATmega32.

(b) Digi® XBee® 1mW PCB radio module.

Figure 4.3: Control and communication devices.

4.3.3 Electronic Devices

Two electronic boards are embedded in the *ErekoBot* σ . The base board contains voltage regulators to provide $3.3\,V$, $5\,V$ and $6\,V$ for the devices in the circuit. A capacitor based filter the signals, resulting in a clean DC signal for the logic inputs. In addition, there are three LEDs to help debugging. The lateral board allows the connection between the electronic board and the sensors. The complete circuit schematics and circuit board drawings are shown in Appendix III.

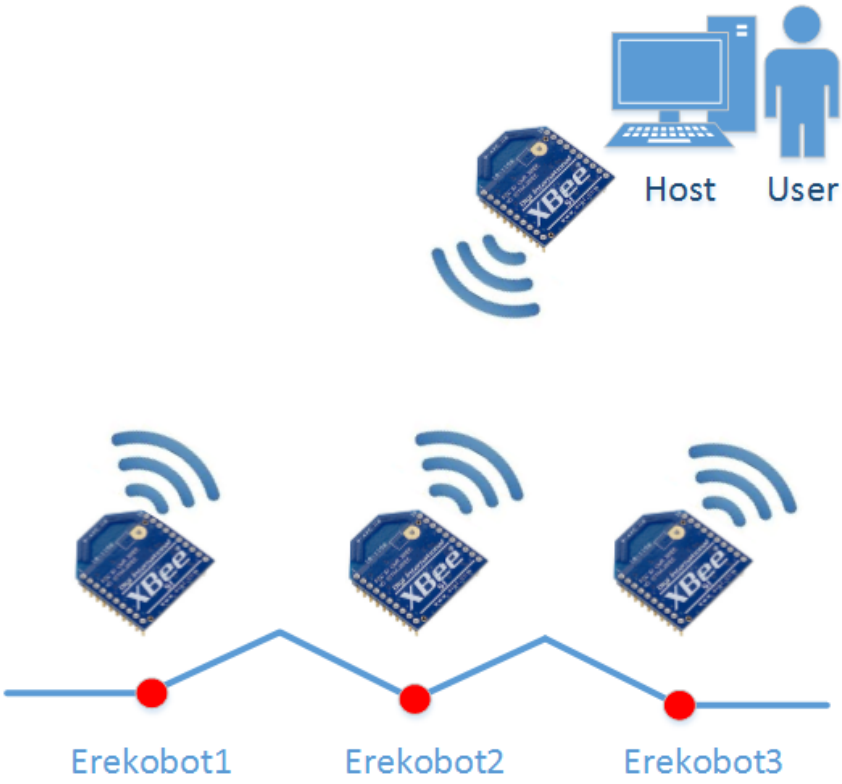


Figure 4.4: Xbee configuration.

4.3.4 Power Supply

There are two options for power supply in modular robotics: *ErekoBot* σ has an external $12\,VDC$ power supply while internal Lithium Polymer batteries are used in *ErekoBot* $\alpha\,v.2$. The choice for the external power supply in *ErekoBot* σ was done so the module maximum weight would not be exceeded (see Section 4.4.4 for more information).

4.4 Mechanical Design

The main idea of ErekoBot σ was that they should not only be easy to build, but at a low cost. Therefore, a simple intermodular connection is adopted (Section 4.4.1) and simple structural pieces are employed (Section 4.4.3). Finally, the maximum weight that the servomotor supports is examined (Section 4.4.4).

4.4.1 Intermodular Connection

The *ErekoBot* σ has a Velcro^{TM} connection for simplicity, which makes the self-reconfigurability impossible, but the manual reconfigurability easy for experimental tests. In the future, with self-reconfigurability it is possible to change the shape of the module inside the pipeline and not only outside by the operator.

4.4.2 Servo Motor

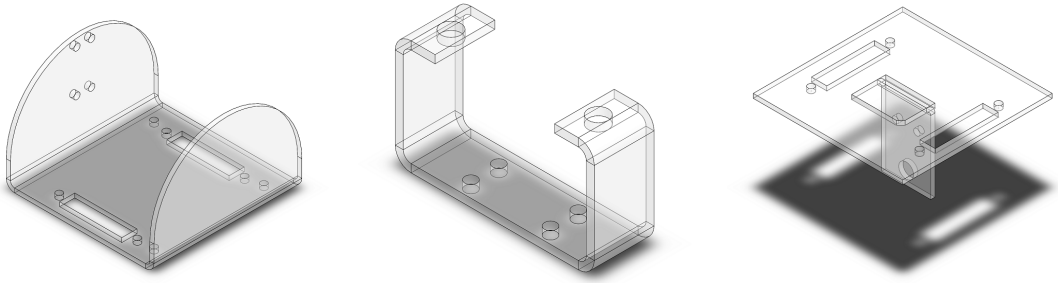
It is decided to use the 3 pole servomotor Hitec HS-85BB (Figure 4.5) with a $3.5\,\frac{kg}{cm}$ (when powered at $6.0\,V$). It is a small ($29 \times 13 \times 30$ in mm) and light ($19\,g$) servomotor with a high torque and strong resin gear train. The power consumption depends on the characteristics of the robot, such as weight, form of connection and number of modules lifted.

Figure 4.5: Hitec HS-85BB.

The Pulse Width Modulation (PWM) sent to the motor determines position of the shaft, and, based on the duration of the pulse sent via the control wire, the rotor will turn to the desired position.

4.4.3 Geometric Description and Materials

ErekoBot σ is a cube with $70\,mm$ edge, formed by an acrylic $2\,mm$ cut by laser. This material proved to be light and resistant to impacts. There are three pieces: the Base, the Motor Holder and the Rod-Cover set (Figure 4.6). The Base holds two IR sensors and the electronic board (manually and soldered manually); the Rod holds the IMU; and the Cover holds the other two IR sensors.



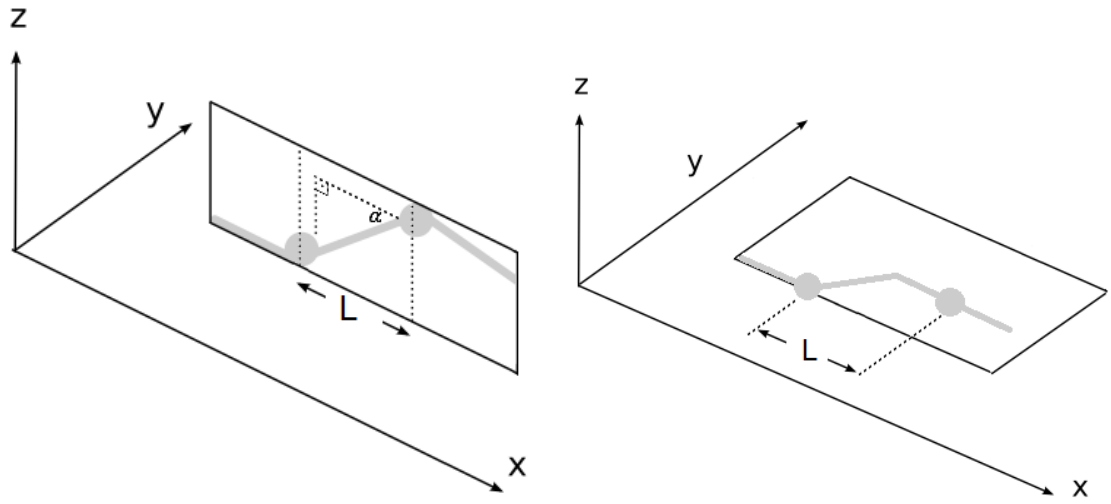
(a) Base. (b) Motor Holder. (c) Rod-cover.

Figure 4.6: *ErekoBot* σ mechanic pieces.

The Rod-cover set is screwed into the servo motor, which is screwed into the Motor Holder, which, finally, is screwed into the Base (Appendix IV presents the plans for the pieces). The only degree of freedom is given by the servo motor that moves the Rod-cover set.

4.4.4 Weight Analysis

For this analysis, lets consider the servomotor and the mass center positioned at the module center. In addition, all modules connections are in the same plane. There are two types of connection that provide two types of movements: in the vertical plane *pitch* (Figure 4.7a) and in the horizontal plane *yaw* (Figure 4.7b).



(a) Pitch. (b) Yaw.

Figure 4.7: Possible connections between two modules.

The torque intensity τ necessary for the module A lift the module B in pitch is

$$\tau_{pitch} = mgL \sin \alpha,$$

where m is the module mass, g is the gravitational force, L is the length of the module (defined in Section 3.2) and α is the pitch angle of the segment related to the vertical axis.

In addition, the torque τ necessary in yaw is

$$\tau_{yaw} = mgL \sin \mu_s,$$

where μ_s is the coefficient of friction in the horizontal plane.

The maximum torque τ_{max} occurs when $\alpha = \pi/2$ or $\mu = 1$, which corresponds in both cases to

$$\tau_{max} = m\,g\,L + m\,g\,(2L) = 3\,m\,g\,L,$$

as can be seen in the free body diagram in Figure 4.8.

The Hitec HS-85BB has a torque $\tau_a = 3.5\,kg/cm$ or $\tau_a = 0.3437\,Nm$. The gravitational force is $g = 9.82\,m/s^2$ and the module length is $70\,mm = 0.070\,m$, therefore, the maximum weight of *ErekoBot* σ can be calculated as:

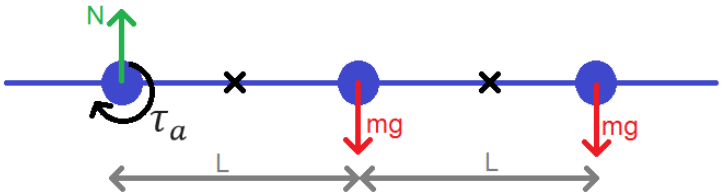


Figure 4.8: Free Body Diagram.

$$0.3437 = 3 * 9.82 * 0.07m$$
$$m = 0.167\,kg$$

Considering a security factor of 25%, the maximum weight of each module is

$$m_{max} = 0.75\,m = 0.12525\,kg = 125\,g. \tag{4.1}$$

4.5 Module Analysis

After manufacturing one *ErekoBot* σ (Figures 4.9 and 4.10), the differences between the actual and the estimated weights are measured (Section 4.5.1). Then, its main characteristics are compared with other SRM modules (Section 4.5.2).

4.5.1 Actual and estimated weights

Each piece of the *ErekoBot* σ was weighted and, then, compared to their estimated values by the data sheets or by *SolidWorks*® in Table 4.1. Although each estimated and measured values varies (18.9%,−20.0%), the total weight of the module is closer to the estimated value: only 0.2% lighter.

Piece	**Estimated** (g)	**Measured** (g)	**Difference**
Base	22.72	28	18.9%
Motor Holder	3.94	3.4	−15.9%
Rod-Cover set	12.46	14.8	15.8%
Electronic Circuit	30	25	−20.0%
Servo Motor	19	19	0.0%
IR Sensors	$4 * 3.5$	$4 * 3.2$	−9.4%
IMU	0.5	0.4	−25%
XBee	10	9.4	−6.4%
Other	5	4.6	−8.7%
Total	117.62	117.4	−0.2%

Table 4.1: Estimated and measured weights.

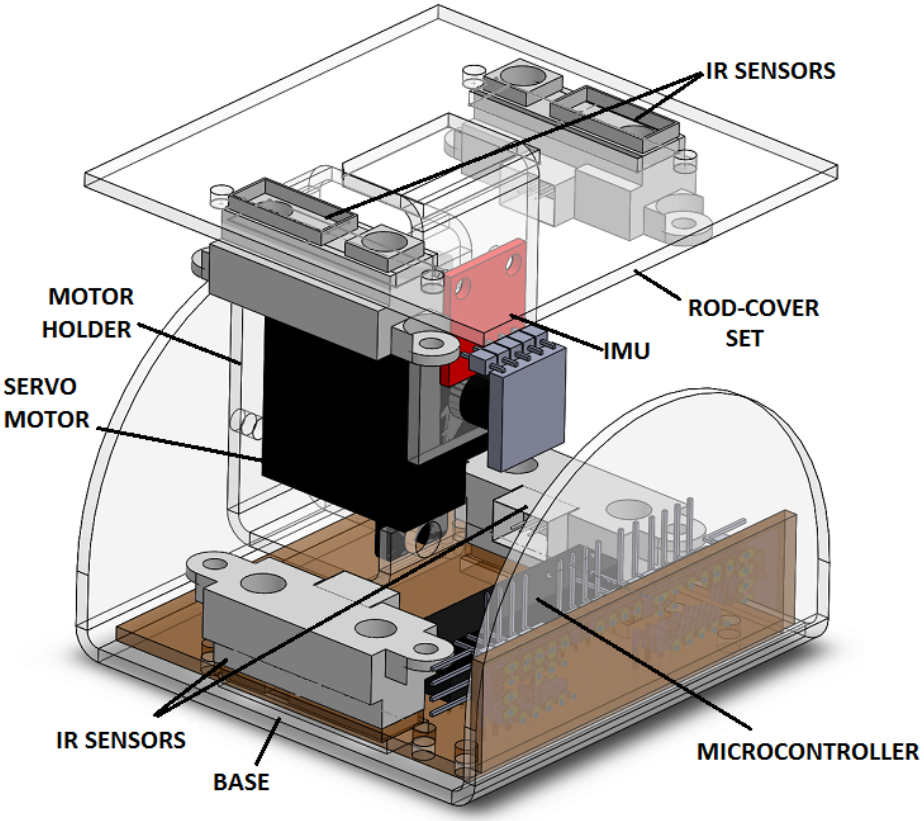


Figure 4.9: Estimated *ErekoBot σ* (drawn in *SolidWorks*®).

4.5.2 Comparison between *ErekoBot σ* and other modular robots

Table 4.2 resumes the comparison between *ErekoBot σ* and the most recent SRM modules presented in Section 2.2.3. All of them use wireless communication in order to decrease dependences with wires in reconfiguration. In addition, most modules also are homogeneous, which eases design, manufacture and simple tests. Except for CoSMO and Roombots, most modules, including *ErekoBot σ* , uses servomotors to enable position control for more complex movements.

In comparison with other modules, *ErekoBot σ* is smaller and lighter, mainly because it has only one degree of freedom and no motor for connection. This selection was made considering the *ErekoBot σ* goals (sensors with simplicity and low cost).

Only Roombots and SMORES do not have internal sensors. In the case of Roombot, users get information about the modules using computer vision with external cameras. The research on SMORES focus on a blind connection using magnets.

The Transmote module just uses IR sensors for a similar alignment of the *ErekoBot σ* , but there is no feedback of internal positions.

CoSMO and UBot have a more complete selection of sensors (orientation and proximity) but CosMO sensors have not been tested yet and UBot design used heterogeneous modules, and just the head of the robot has these sensors.

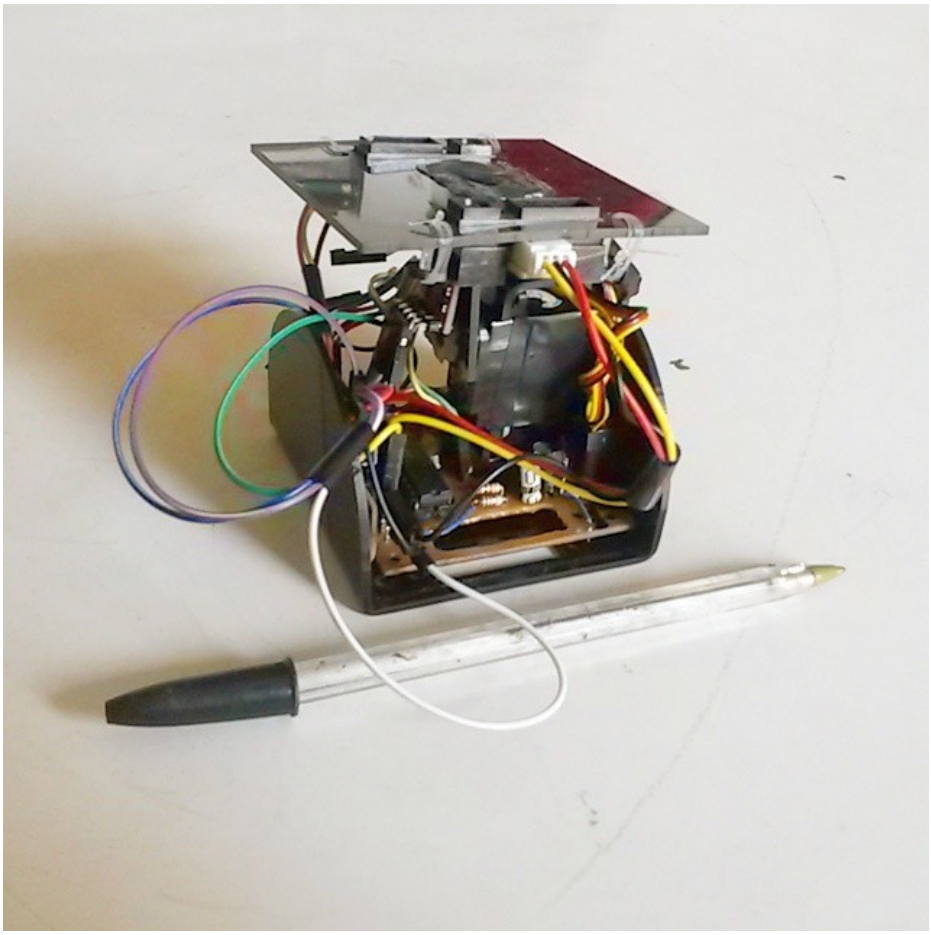


Figure 4.10: Actual *ErekoBot* σ .

Name	***ErekoBot* σ**	**SMORES**	**UBot**	**CoSMO**	**Roombots**	**Transmote**
Type	Homogeneous	Homogeneous	Heterogeneous	Homogeneous	Homogeneous	Homogeneous
Dimensions (mm)	$70 \times 70 \times 70$	$100 \times 100 \times 90$	$80 \times 80 \times 80$	$70 \times 70 \times 70$	$100 \times 100 \times 90$	$80 \times 80 \times 80$
Weight (g)	118	520	280	1250	1400	Not informed
CPU	ATMega32	MBED	Not informed	Dual Core Blackfin, MSP430 and CortexM3	MIPS dsPIC33	RISC core
Motors	Servo	Servo	Servo	DC	DC	Servo
DoF	1	4	2	1	3	3
Connectors	Manual by velcro	Active and passive pins	Active and passive pins	Active and passive pins	Active and passive pins	Active and passive pins
Communication	Wireless	Wireless	Wireless	Wireless	Wireless	Wireless
Sensors	IMU and IR sensors	No sensors	Wireless vision sensor, accelerometer, IR range finder and linear Hall Sensor	Hall sensor, IMU, current sensor, camera and IR sensors	No sensors	IR sensors

Table 4.2: Comparison between *ErekoBot* σ and other SRM modules.

4.6 Conclusions

In this chapter, all the devices and architectures of *ErekoBot* σ were defined in order to meet its desired requirements. In Section 5, the manufactured module is used in experiments that are described and analysed.

Chapter 5

Experiments and Results

This chapter describes the platform and setup developed to carry out the experiments (Sections 5.1.1, 5.2.1 and 5.3.1). Then, the results are presented in sections 5.1.2, 5.2.2 and 5.3.2.

The experiments described in this chapter aim to verify the quality of the information obtained with the data acquired from the sensors. Three types of experiments are performed: orientation experiment, which employs the IMU acquired data; distance experiment, that applies the data coming from the IR sensors; and the alignment experiment, for positioning the module face parallel to a detected obstacle.

5.1 Orientation Experiment

The main goal of these experiments is to estimate the *ErekoBot* σ position using the accelerometer or the gyroscope from the IMU. In these experiments, the microcontroller reads the IMU data and sends them thought the Digi® XBee® 1mW PCB radio module to the host computer, which filters the signal using the Complementary Filter described in Section 2.3.4.1.

The IMU readings were tested in two situations: when the module is at rest position and when the module is in movement. The experiments are considered successful if it is possible to estimate the pitch angle of the module without deviation.

The modules have only one degree of freedom, that can be arranged to perform yaw or pitch angles (see Figure 5.1). As it is not possible to make the module perform a roll movement, the roll angle will not be analyzed here.

In addition, because of the Gimbal Lock (Section 2.4.1.1), it is not possible to estimate the yaw position with the Complementary Filter combining information from the accelerometer and gyroscope sensors. Therefore, the only angle to be analyzed is the pitch angle.

The complementary filter is adjusted in the host and then, the result of P_{err} is used in each module.

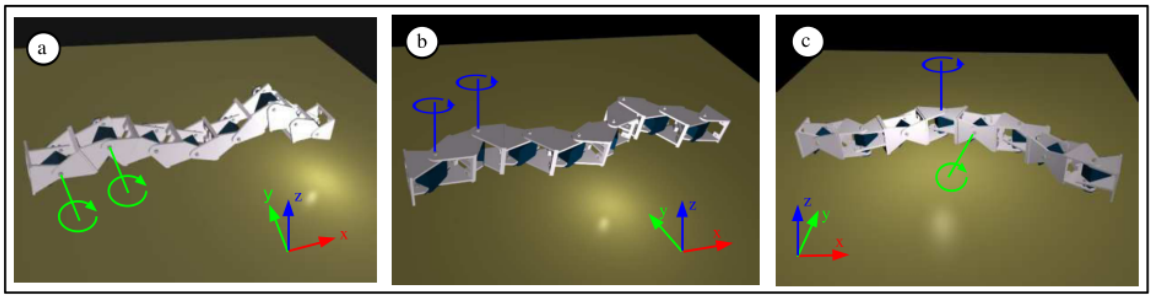


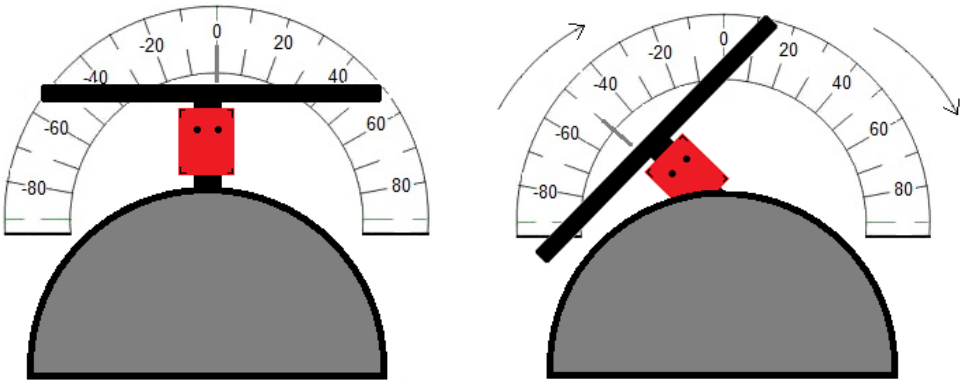
Figure 5.1: Different types of connections in modular robots a) Pitch-pitch. b) Yaw-yaw. c) Pitch-yaw.

5.1.1 Experimental Method

Figure 5.2a shows the position and setup for the first experiment. The module rests at position 0° and the IMU measurements are read for 80 seconds with a sample time of 200 ms. Therefore, the module send 400 measurements for the host, that calculates offline the filtered signals with the Complementary Filter algorithm.

Figure 5.2b shows how the module moves from position $-45°$ to 45°. The algorithm employed by the module consists of the following steps:

1. Read the IMU 10 times;
2. Send the measurements to the host;
3. Wait 200 ms;
4. Advance 5°.
5. If position is less than 45°, go back to 1



(a) Rest position. (b) Moving position.

Figure 5.2: Positions of *ErekoBot* σ for the orientation experiment.

5.1.2 Results

Using the Equations (3.2), (3.12), (3.13), the host computed the pitch angle from the IMU measurements, while the module was holding at rest (0°), as pictured in Figure 5.3. It can be seen that the results based on the accelerometer provided exact angle values: even if the mean value varies in time, it remains close to $-3.30°$ with a standard deviation of 12.74°. However, these values are inaccurate because there is a great variation at each measurement. In contrast, the gyroscope based angles were more accurate, but inexact due to the deviation that occurs over time.

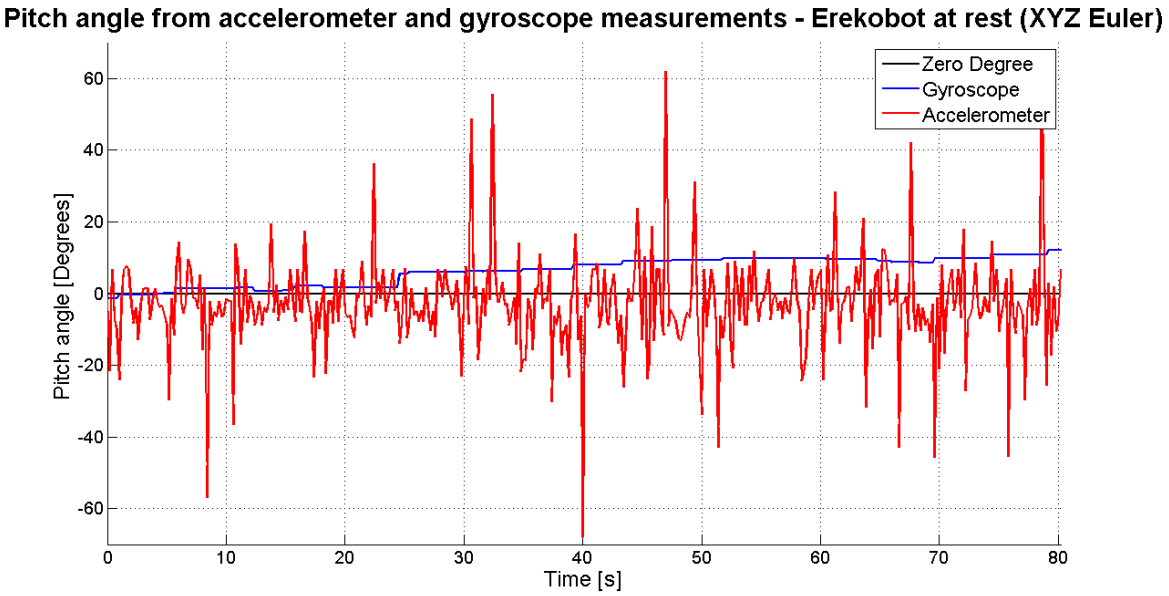


Figure 5.3: Pitch angle estimations of position 0° without a filter.

The findings above imply that accelerometer and gyroscope data are complementary which asserts that the Complementary Filter can be used for combining their measurements and achieving a better estimation of the module pitch angle. The results of pitch angle estimation using the Complementary Filter are presented in Figure 5.4. As mentioned in Section 3.3.1.3, when P_{err} is close to zero, the Complementary Filter eliminates the accelerometer readings (shown as darker lines) and when P_{err} is close to one, the Complementary Filter does not take into account the gyroscope measurements.

The pitch angle was also calculated by the host computer from the IMU measurements when the motor is moving from $-45°$ to 45°, and the readings are shown in the graph of Figure 5.5. These results are similar to the graph of Figure 5.3: the accelerometer provided exact but inaccurate angles while the gyroscope supplies accurate but inexact angles. Figure 5.6 presents the graph with the estimation obtained using the Complementary Filter, when P_{err} is 0.9, 0.7, 0.5, 0.3 and 0.1.

The results from Figures 5.4 and 5.6 provide enough evidence towards employing the complementary filter for future estimations with $P_{err} = 0.5$ because it is the outcome that best approximates the straight line $-45°$ to 45°.

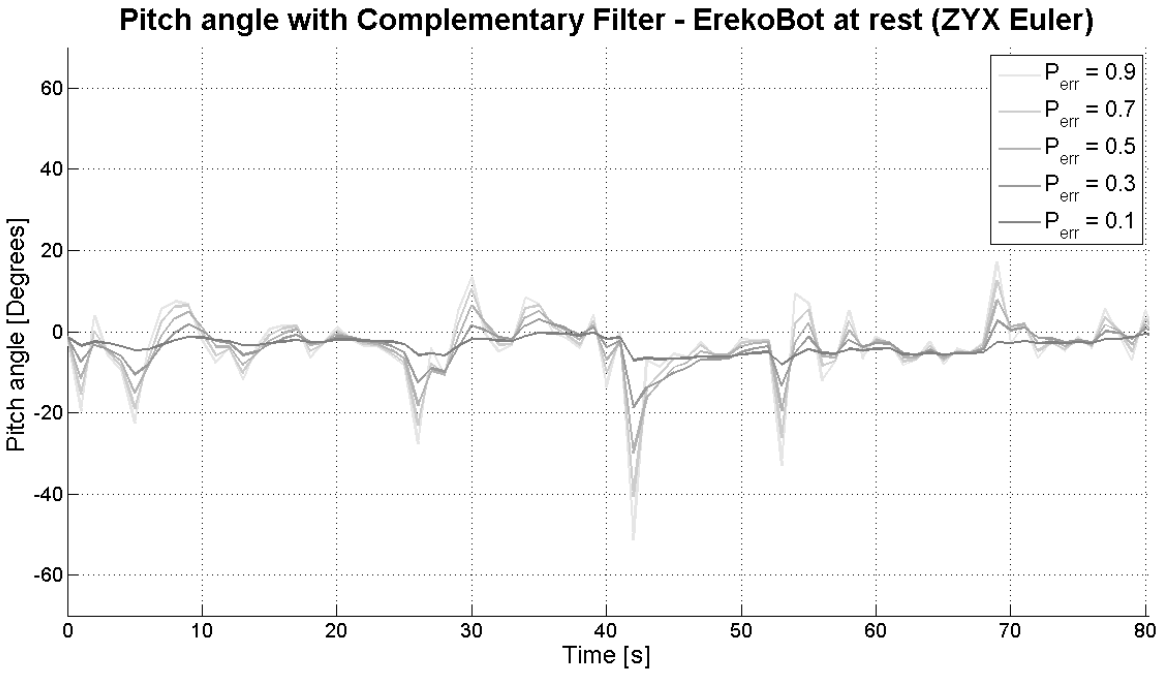


Figure 5.4: Pitch angle estimations of position 0° filtered with $P_{err} = 0.9$, $P_{err} = 0.7$, $P_{err} = 0.5$, $P_{err} = 0.3$ and $P_{err} = 0.1$.

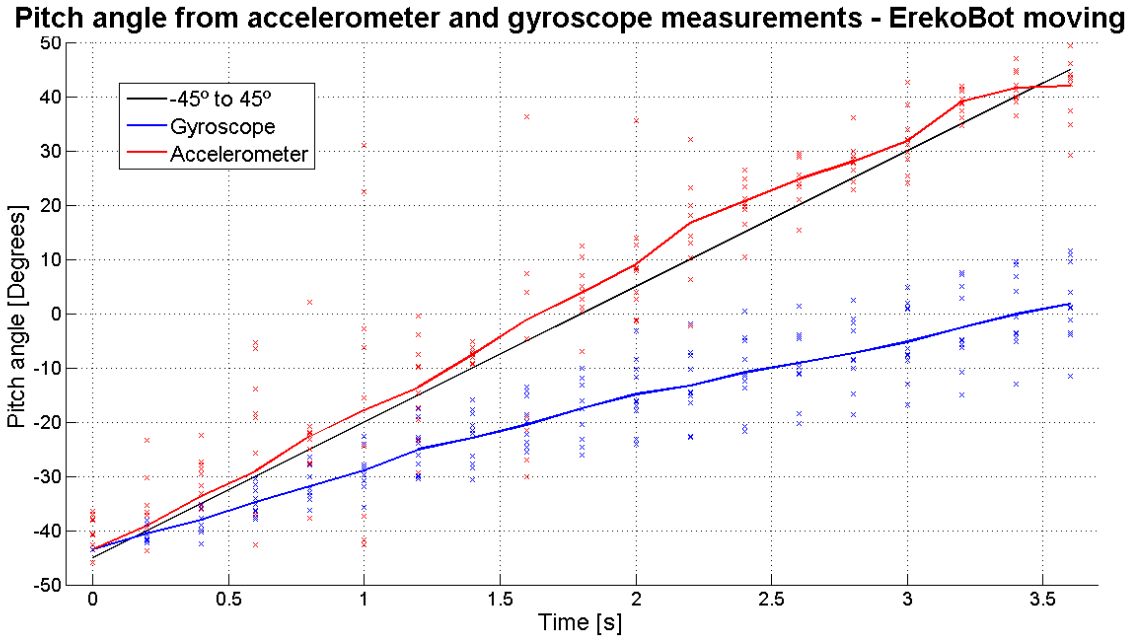


Figure 5.5: Pitch angle estimations of positions −45° to 45° without a filter.

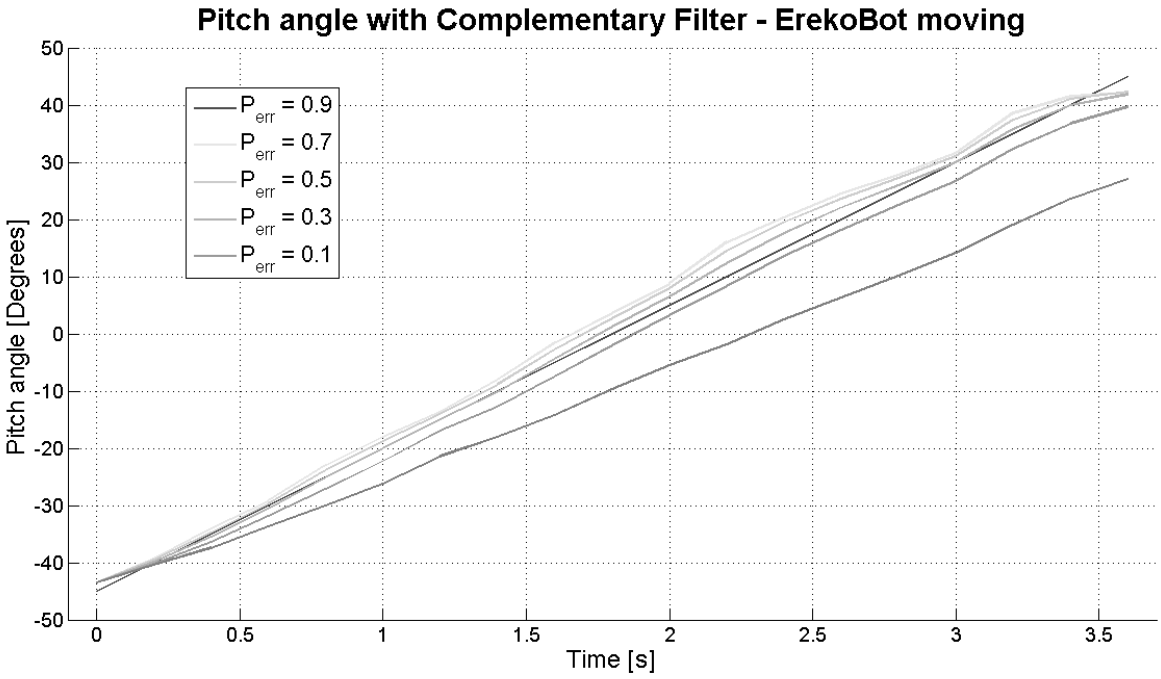


Figure 5.6: Pitch angle estimations of positions $-45°$ to $45°$ filtered with $P_{err} = 0.9$, $P_{err} = 0.7$, $P_{err} = 0.5$, $P_{err} = 0.3$ and $P_{err} = 0.1$.

5.2 Distance Experiment

The main goal of this part of the experiments is to detect the presence of an obstacle and estimate its distance from the *ErekoBot* σ face where the IR sensors are fixed. The dependence between voltage output and estimated distance for the IR Sensors is not linear which suggests the need to calibrate the IR SHARP GP2Y0A41SK0F (Section 2.3.1). After calibration, a lookup table is generated and it is inserted in the microcontroller to be used when estimating the distance between the module face and an obstacle in the environment.

The experiment succeeds when the *ErekoBot* σ is capable of detecting an obstacle and estimating its distance using the lookup table.

5.2.1 Experimental Method

The IR SHARP GP2Y0A41SK0F sensor detects obstacles and measures distances within a range from 40 to 300 mm. In order to assure the correct detection and measurement, the IR Sensor was calibrated by analyzing the non-linear voltage curve detecting a gray obstacle between 10 to 350 mm with a resolution of 10 mm. Figure 5.7 shows the experiment arrangement.

In order to consider possible hysteresis and environmental changes, the measurements were made in both directions. Initially, the sensor is located at 10 mm, then the IR sensor moves slowly for others measures until it reaches 350 mm. At that point, the sensor is moved backwards and the measurements are made again. To make sure the results are redundant, the measurements are

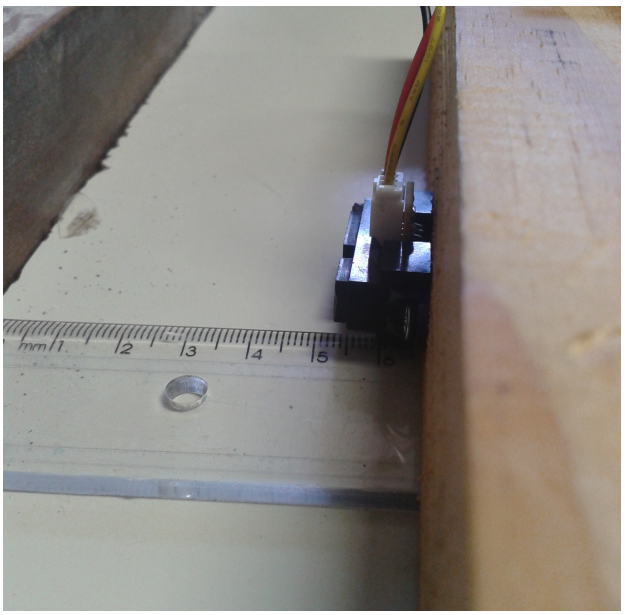


Figure 5.7: Preparation for the distance experiment. The IR Sensor measures the voltage in different distances for calibration.

repeated four times for each position.

A curve based on the sensor readings is built, and the acceptable range is determined. The best power equation is chosen with the help of the computational engine Curve Fitting Toolbox® from MATLAB®.

5.2.2 Results

Figure 5.8 shows the curve obtained by the measurement average values. The dependence between voltage output and proximity distance for IR Sensors is nonlinear, and rather similar to the curve in the IR sensor data sheet (SHARP, 2002). From this curve, it is possible to specify a working range between 5 and 28 cm .

With the Curve Fitting Toolbox® from MATLAB®, the best power equation fit was found

$$V_{IR} = 112.3\, d^{-0.9915},$$

where V_{IR} is the output voltage in Volts and d is the distance, in millimeters, between the IR sensor and the obstacle.

In order to build the lookup table, it is necessary to compute the distance based on the voltage measurement, the Equation (5.2.2) is inverted, which results in

$$d = \left(\frac{112.3}{V_{IR}}\right)^{1.0086}. \tag{5.1}$$

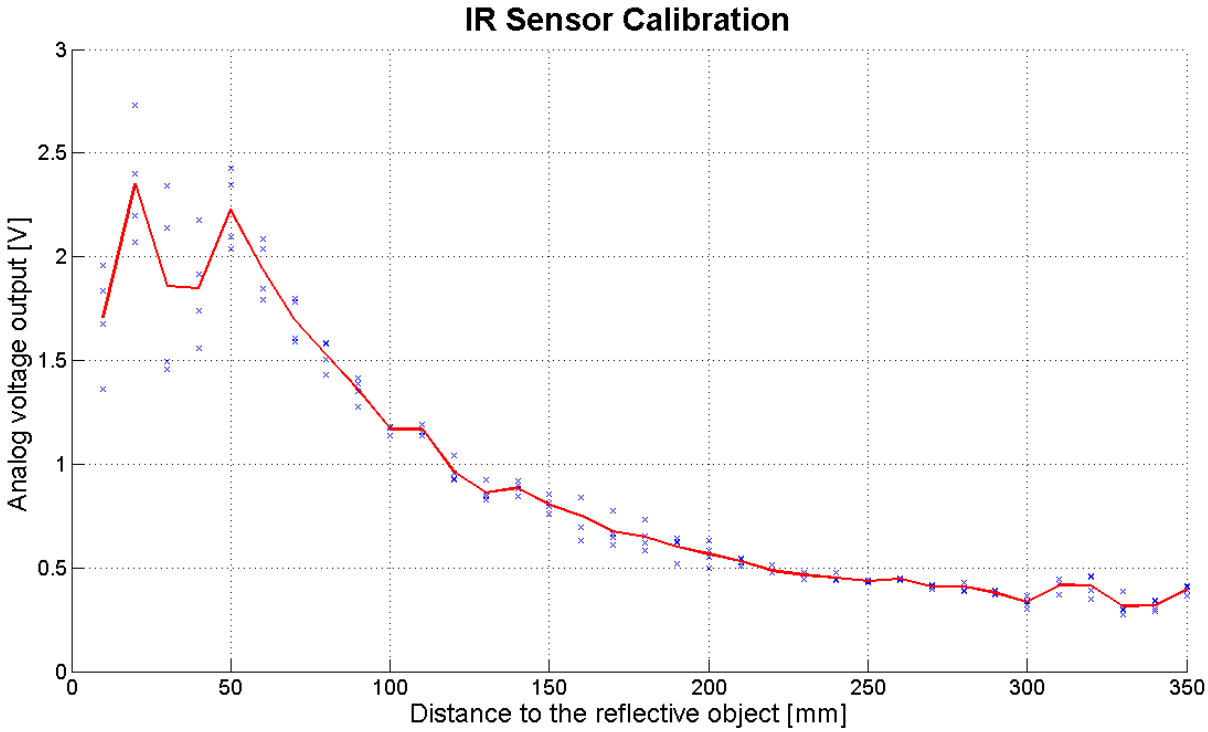


Figure 5.8: IR sensor voltage outputs points (blue markers) and curve formed from the average values (red line).

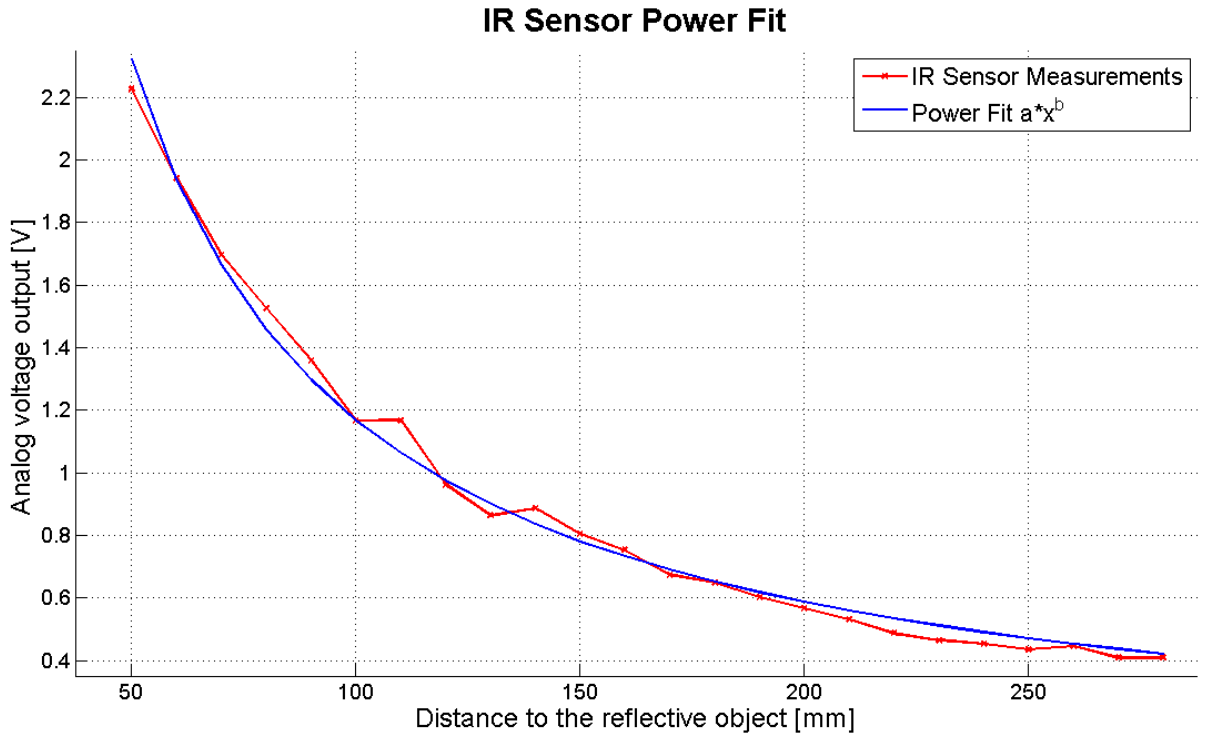


Figure 5.9: IR sensor voltage outputs and power fit from Equation (5.2.2) curves.

5.3 Alignment Experiment

The main goal of this experiment is to detect the presence of an obstacle, estimate its distance from the *ErekoBot* σ face, and align the module sensor surface with the obstacle. In order to verify the Alignment Algorithm from Section 3.4, the algorithm was tested with different planes at different distances.

5.3.1 Experimental Method

Combining the results from Sections 5.1.2 and 5.2.2, the following Alignment Algorithm was applied:

1. The *ErekoBot* σ reads the orientation of the rod at time t ($orient_t(6)$) and calculates the pitch angle using a complementary filter with $P_err = 0.5$;
2. The *ErekoBot* σ reads the objects distance at time t ($proxim_t(4)$) using a lookup table from Equation (5.1);
3. Through Digi® XBee® 1mW PCB radio module , it sends $orient_t(6)$, $proxim_t(4)$ and t to the host computer;
4. If the measured distances are different, the servo motor rotates to an angle θ proportional do the difference measured from IR sensors, in order to align with the plane;
5. After adding 1 to t and waiting for $300\,ms$, the *ErekoBot* σ returns to step 1.

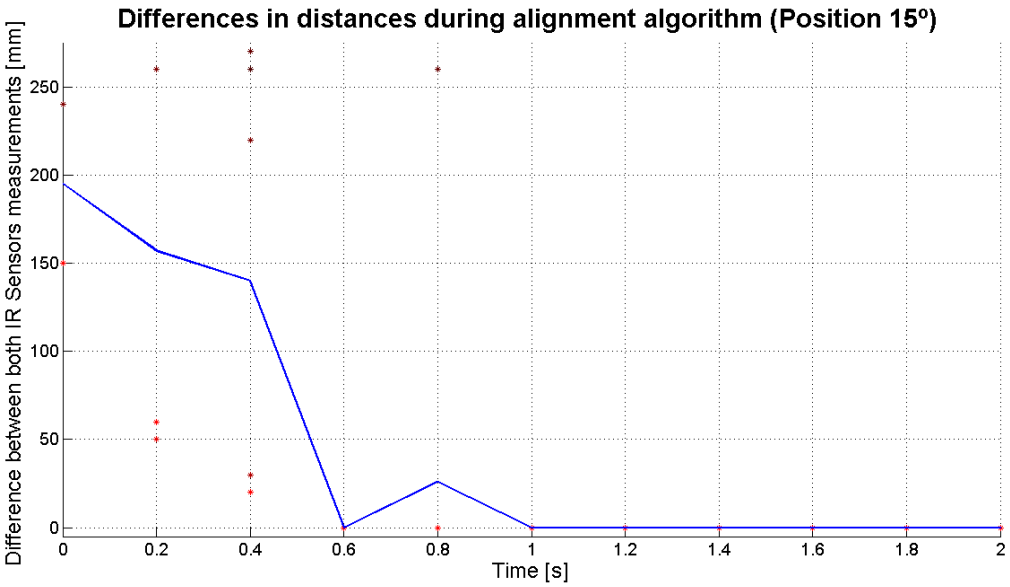
The alignment was tested with three different surfaces: a white sheet of paper, a card paper plane and a black sheet of paper in different positions. The purpose is to test if the alignment works with different surface colors (white, gray and black) and to estimate how long it takes for the *ErekoBot* σ to align with the plane.

5.3.2 Alignment Results

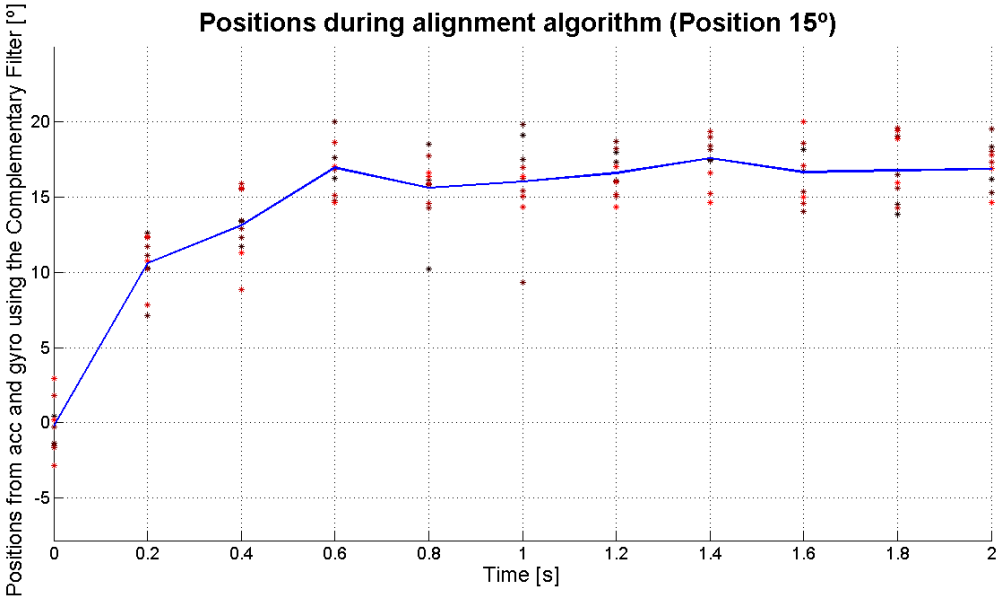
Figures 5.13, 5.14 and 5.15 present the alignment experiments. The results were similar with the three different surfaces tested: the *ErekoBot* σ aligned with the plane. White surfaces reflect better than black surfaces, and, as IR sensors works with infrared reflected light (see Section 2.3.3), the white plane works faster than gray or black planes.

Concerning quantitative results, three tests are shown below. The *ErekoBot* σ executed the alignment algorithm in front of a gray surface, and the differences between the measured distances from the IR sensors and the orientation positions from the IMU (after execution of a complementary filter) are sent to the host. The tests were performed ten times at each position.

In the first test, the *ErekoBot* σ aligned itself with the plane at position 15° in less than 1.0 second, as shown in Figures 5.10a and 5.10b. The servomotor rotates towards positive angles until the module rod gets close to the pitch angle 16°.



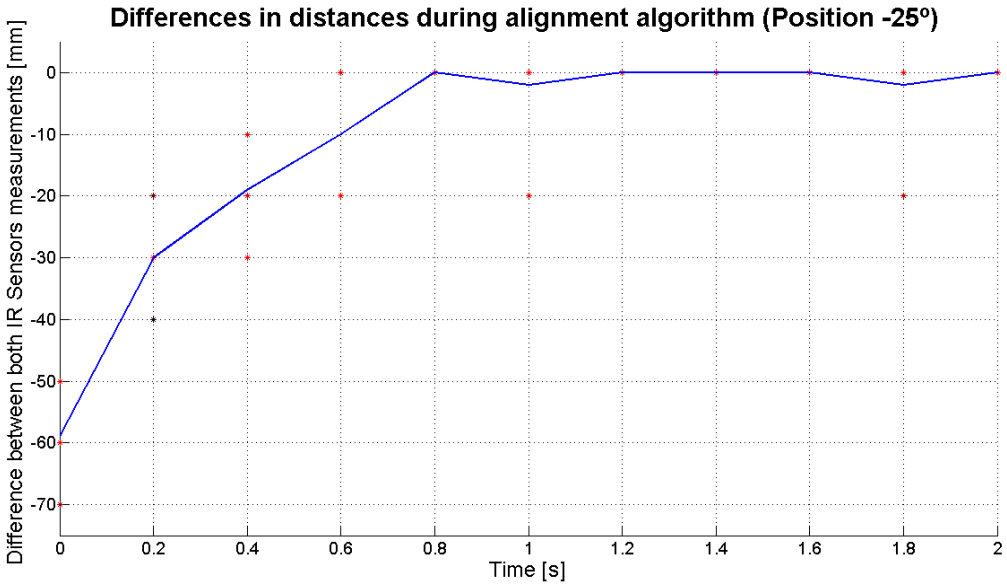
(a) Differences in measured distances during alignment algorithm (Position 15°).



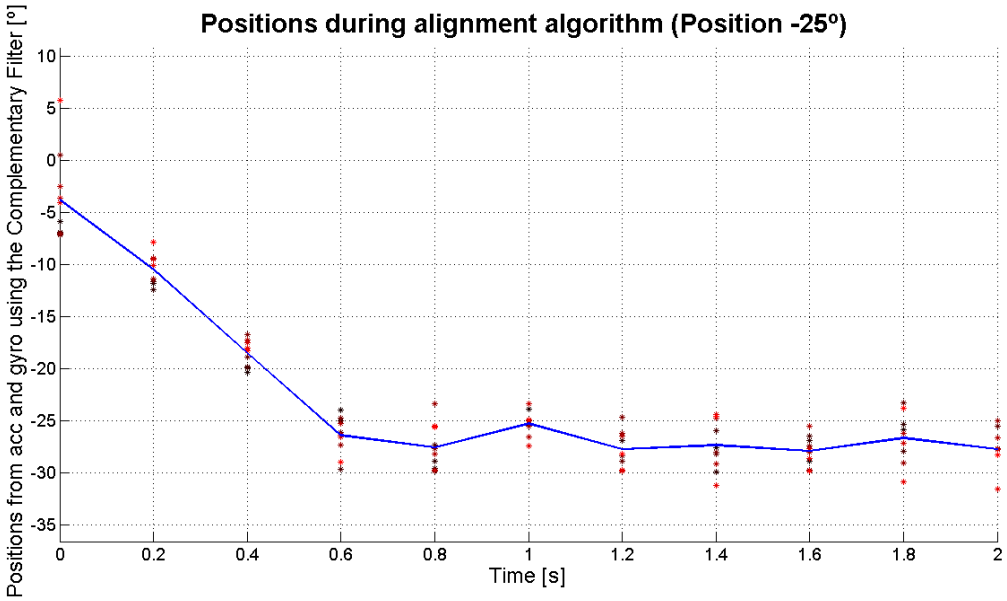
(b) Orientation pitch positions during alignment algorithm (Position 15°).

Figure 5.10: Alignment results for position 15°.

As in the second test, the *ErekoBot* σ could align with the plane at position $-25°$ in less than 1.0 second, as shown in Figures 5.11a and 5.11b. The servomotor rotates towards negative angles until the module rod angle converge towards 27°.



(a) Differences in measured distances during alignment algorithm (Position $-25°$).



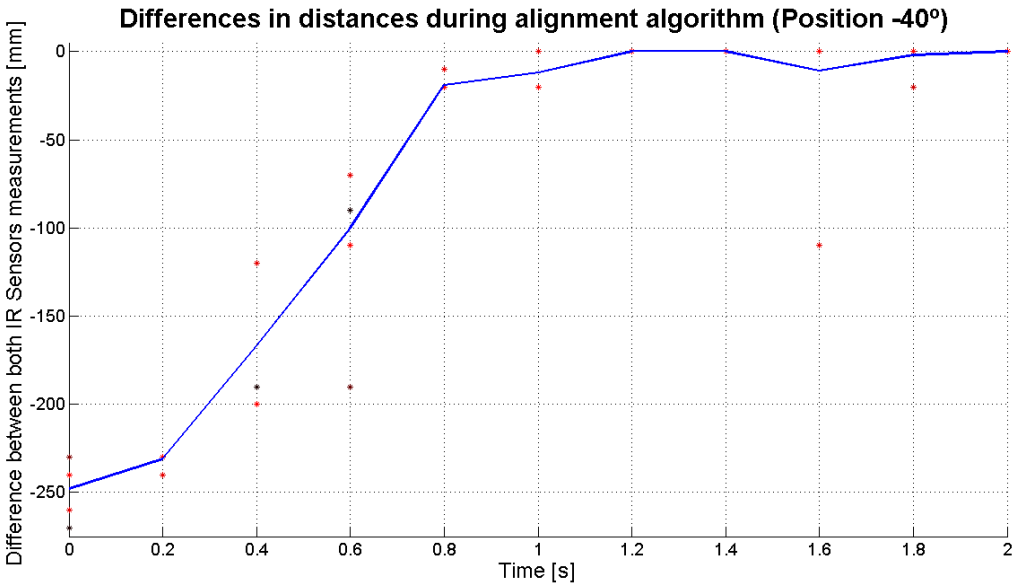
(b) Orientation pitch positions during alignment algorithm (Position $-25°$).

Figure 5.11: Alignment results for position $-25°$.

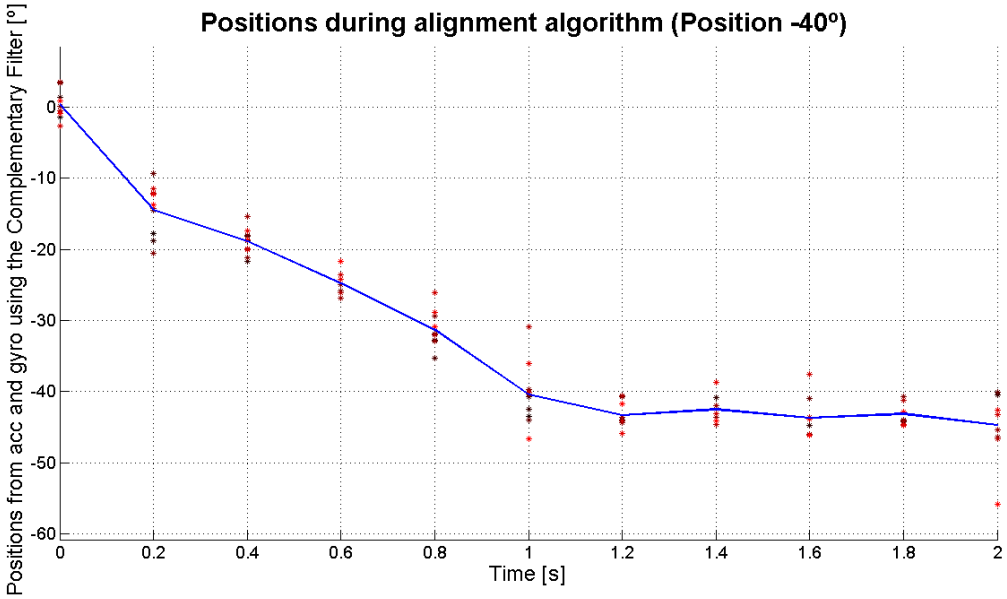
In the third and last test, the *ErekoBot* σ achieved alignment with the plane at position $-40°$ in less than 1.2 second, as shown in Figures 5.12a and 5.12a. The servomotor rotates towards negative angles until the pitch angle value of $-43°$ is attained.

The tests above pointed some limitations:

- Light: occasionally, when there is a direct light pointing to an IR sensor, the measurement



(a) Differences in measured distances during alignment algorithm (Position $-40°$).

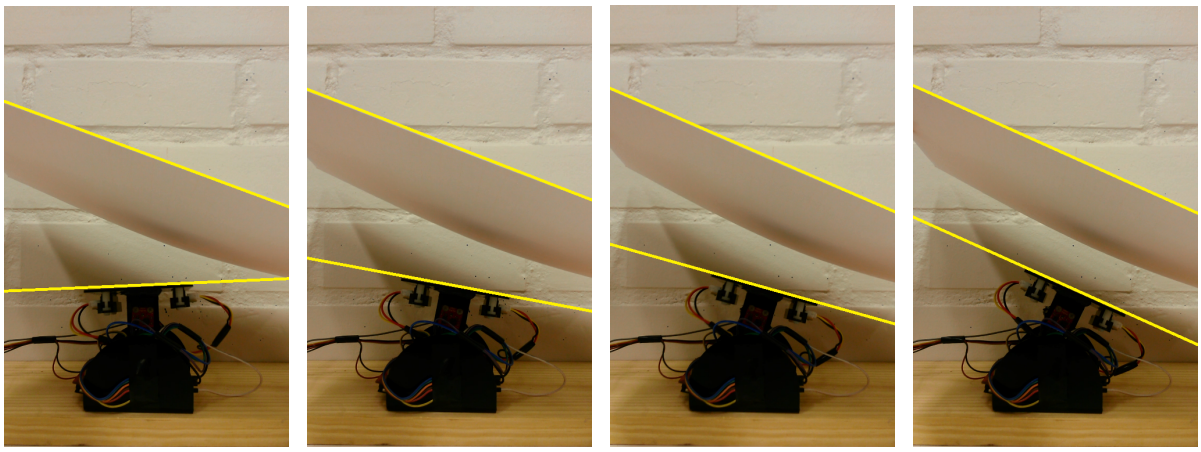


(b) Orientation pitch positions during alignment algorithm (Position $-40°$).

Figure 5.12: Alignment results for position $-40°$.

responses are affected. The IR sensor considers that an object exists at its front and the alignment does not work properly.

- Distances higher than 280 mm: as expected, the IR sensor does not properly measure distances higher than the maximum of the range, 280 mm. The target is considered to be at a distance 280 mm for the alignment algorithm.
- Distances smaller than 50 mm: as expected, the IR sensor does not properly measure distances smaller than the minimum of the range, 50 mm. Observing the graph of Figure 5.8, it is possible that the *ErekoBot* σ consider the object in a position inside the range 50 mm to 280 mm. For future work, an algorithm that consider statistics based on previous distances can be developed to work in such cases.
- Servomotor position higher than 35° and smaller than 145°: in order to avoid the collision of the module internal parts, a maximum and minimum servomotor positions, from 35° to 145°, were determined.
- The calibration for the IR sensor in Figure 5.9 works just for a gray obstacle, and different colors may produce different lookup tables. However, as the *ErekoBot* σ aligns based on differences between these distances, the alignment still works with an homogeneous plane (same color and texture).



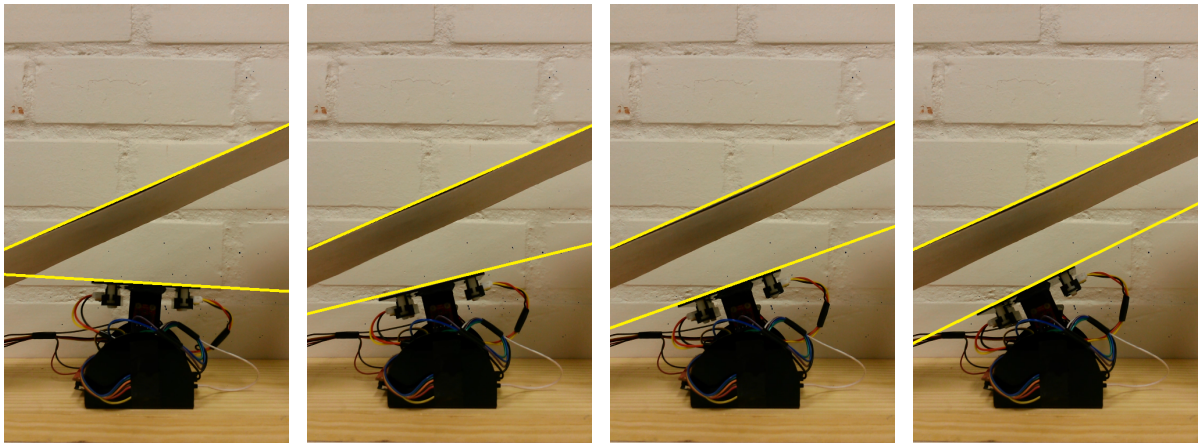
(a) 0 second. (b) 0.2 second. (c) 0.4 second. (d) 0.6 second.

Figure 5.13: Alignment results with a white sheet of paper plane (white).

Considering these limitations it is not possible to confirm that the robot can be used in pipelines yet.

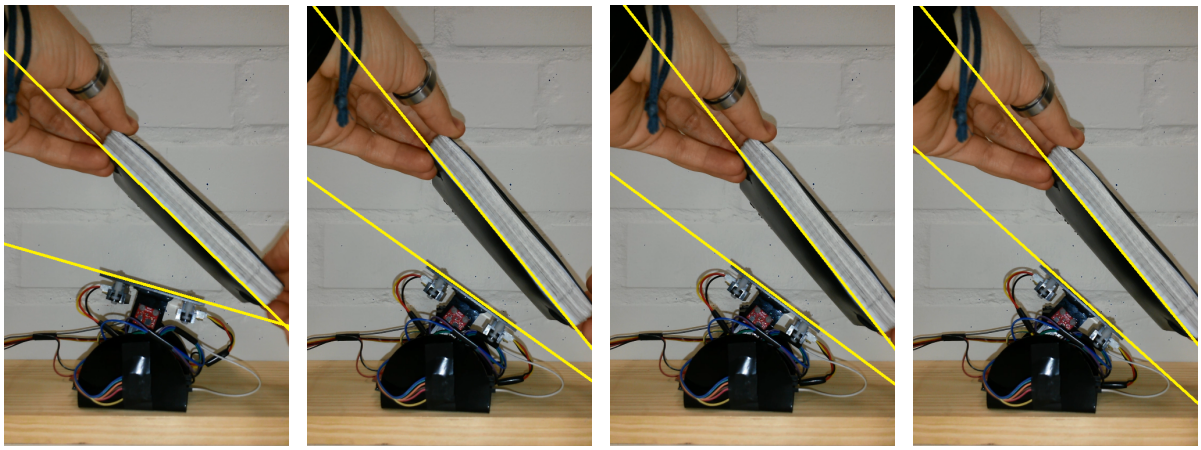
5.4 Conclusions

A SRM module was designed and manufactured respecting the conceptual design proposed in Chapter 4, which allowed us to test the models from Chapter 3. The embedded sensors are used for:



(a) 0 second. (b) 0.2 second. (c) 0.4 second. (d) 0.6 second.

Figure 5.14: Alignment results with a card paper plane (gray).



(a) 0 second. (b) 0.2 second. (c) 0.4 second. (d) 0.6 second.

Figure 5.15: Alignment results with a black sheet of paper plane (black).

1. Estimating the orientation pose of *ErekoBot* σ using the Complementary Filter.

2. Calibrating the IR sensors for distance measurements.

3. Aligning the *ErekoBot* σ face with a detected plane.

With the results of the IMU tests (Figure 5.3 and Figure 5.5), it is possible to verify that the sensors readings to estimate the module orientation are either noisy or present drift over time. To overcome this problem, a complementary filter was employed, which decreases noise and drift, estimating faster and not overloading the module processor. Therefore, for faster and more accurate stabilization of the *ErekoBot* σ , the complementary filter is chosen as the default filter used for future processes using the IMU.

The IR sensor calibration provided a similar curve to the one that appears in the data sheet (SHARP, 2002), and the results from Equation (5.1) were converted in a LUT programmed in the Atmel® AVR® ATmega8 .

With these results, it was possible to perform the alignment, and demonstrate the viability of the algorithm described in Section 3.4. With the alignment experiment results, it is possible to confirm that the robot may not be used for pipeline inspection applications yet.

Chapter 6

Conclusions

This Chapter first presents the principal contributions of this dissertation with a brief discussion of its implications and, then, indicates some of possible future lines of investigation to this work.

6.1 Main Contributions

The *ErekoBot* σ is a simple reconfigurable module with sensors able to react to the environment based on sensors response. In this work, the module was designed, manufactured and tested, which allowed us to answer the questions presented in Section 1.2:

1. *What kind of sensors are more common in recent SRM robotics?* Table 4.2 resumes the main characteristics of five of the most recent modules. Roombots and SMORES do not have internal sensors; Transmote just uses proximity sensors; and CoSMO and UBot have a more complete selection of orientation and proximity sensors.

2. *What kind of requirements does the robot have?* Section 3.2 presents three main requirements for the *ErekoBot* σ : the abilities to (1) detect an obstacle (Figure 3.3a), (2) align with a plane simulating a pipeline (Figure 3.3b) and (3) estimate its own pose (Figure 3.3c).

3. *What kind of sensors are going to be used in this module?* Section 4.2 presents the sensors selection. The *ErekoBot* σ has four Infrared Sensors (GP2Y0A41SK0F) for obstacles detection and the alignment algorithm. In addition, *ErekoBot* σ has an IMU (with ADXL345 and ITG-3200) for rotation motion measurements.

4. *What is the shape, size, weight, circuits and materials of the module?* Chapter 4 focus on the module design, and Figures 4.6, 4.9, 4.10 and Table 4.2 illustrates and resumes shape, size, weight, circuits and materials of the complete module.

5. *Does the chosen sensors are sufficient for our needs?* Chapter 5 presents the results of the experiments. The *ErekoBot* σ is able to: (1) detect an obstacle between 5 and 28 cm (Figure 5.9), (2) align with a plane simulating a pipeline (Figures 5.13, 5.14 and 5.15), and

(3) estimate its own pose with a Complementary Filter from gyroscope and accelerometer measurements (Figures 5.4 and 5.6).

After answering these initial questions, the *ErekoBot* σ presented limitations:

1. *Yaw angle* With just accelerometer and gyroscope is not possible yet to measure the yaw angle.
2. *Velcro*^{TM} The Velcro$^{\text{TM}}$ limits the reconfigurability of the module.
3. *Cable* The power cable limits the locomotion of the robot.
4. *Alignment limitations* Light, distances, servomotors positions and calibration of the IR sensors are described in Section 5.3.2.

The designed module shows evidence towards its application in pipeline inspections, however the proposed solutions have its limitations and are not yet viable for field application.

6.2 Future Lines of Investigation

Some lines of investigation are presented to give continuity to this dissertation (the order that appears is an author suggestion of importance):

- *Thinning the module* It is still possible to thin the module, changing the acrylic for a lighter material (such as resin, carbon fiber) and changing the printed circuit board (PCB) for a surface-mount device (SMD).
- *Cable/Energy* Thinning the module allows the use of an autonomous energy source, eliminating the cable requirement of the system.
- *Magnetometer* For future designs, the magnetometer can be used to determine the yaw rotation.
- *Alignment and obstacle deviation algorithms* This work presents codes and algorithms for reading the sensors values and converting them to system more familiar to users (distance in millimeters, orientation in vectors). Nevertheless, control laws were not added to find the best response for the modules. It is possible to make a close control loop.
- *Connection* Another line of investigation regards connection between modules, it is fundamental to substitute the Velcro$^{\text{TM}}$ for a male-female system with DC motors in each face of the module.
- *Computational vision method* The homogeneity of *ErekoBot* σ gives advantage for a modules series production. However, one of the modules may carry a camera to allow the use of computational vision methods for obstacles detection and alignments.

- *Distributed Robotic systems* After solving the alignment system, it is possible to work with distributed robotic systems, analyzing when to connect, how to connect and what kind of collaborations between the fleet will be established.

Although there are several limitations in this system, this module design and tests contribute to the modular robotics in order to build a versatile system to change automation of maintenance and leaks detection processes in the future.

6.3 Publications

The work in this manuscript allowed the development, directly or indirectly, of the following papers accepted for presentation at conferences:

- Kinematics of a Hexapod Modular Robot (GONCALVES et al., 2013).
- Data Communication and Protocols in self-reconfigurable modular robots (SOUSA; VIANA; KOIKE, 2013a).
- Programando a movimentação em robôs modulares autorreconfiguráveis ápodes para inspeção em tubulações (SOUSA; VIANA; KOIKE, 2013b).
- Sensors in Reconfigurable Modular Robot for Pipeline Inspection : Design and Tests of a Prototype (SOUSA; VIANA; KOIKE, 2014).
- ErekoBot Sigma: Protótipo de um robô modular reconfigurável com sensores (SOUSA et al., 2014).

Bibliography

Analog Devices Inc. *Digital Accelerometer ADXL345 Datasheet*. [S.l.], 2009. Cited on page 39.

ARCHILA, J. F.; BECKER, M. Study of Robots to Pipelines, Mathematical Models and Simulation. In: *2013 Latin American Robotics Symposium and Competition*. IEEE, 2013. p. 18–23. ISBN 978-0-7695-5139-5. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6693264http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6693264>. Cited 2 times on pages vi e 5.

ATMEL. *ATmel AVR Atmega32 Datasheet*. [S.l.], 2011. Cited on page 40.

BANKS, E. R. Universality in cellular automata. In: BURKS, A. W. (Ed.). *11th Annual Symposium on Switching and Automata Theory (swat 1970)*. IEEE, 1970. p. 194–215. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4569649>. Cited on page 8.

BAUER, J. *Milling Robot for processing the internal walls of inaccessible pipelines*. 2011. Disponível em: <http://www.google.com/patents/US8573889>. Cited on page 6.

BENTLEY, J. P. *Principles of Measurement Systems*. 4th. ed. Harlow: Pearson Education, 2005. ISBN 9780130430281. Cited 6 times on pages xiv, 1, 13, 14, 15 e 22.

BONARDI, S. et al. Collaborative manipulation and transport of passive pieces using the self-reconfigurable modular robots roombots. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013. p. 2406–2412. ISBN 978-1-4673-6358-7. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6696694>. Cited 3 times on pages vi, 11 e 12.

Central Intelligence Agency. *The World Factbook 2013-4*. Washington, DC, 2013. Disponível em: <https://www.cia.gov/library/publications/the-world-factbook/fields/2117.html>. Cited on page 4.

CHOI, H.; RYEW, S. Robotic system with active steering capability for internal inspection of urban gas pipelines. *Mechatronics*, v. 12, n. 5, p. 713–736, jun. 2002. ISSN 09574158. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0957415801000228>. Cited 2 times on pages 5 e 6.

CUI, X. et al. A homogenous CPG-network for multimode locomotion control of modular self-reconfigurable robot. In: *2012 IEEE International Conference on Mechatronics and Automation*. IEEE, 2012. p. 2526–2530. ISBN 978-1-4673-1278-3. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6285744>. Cited 2 times on pages vi e 10.

DAVEY, J.; KWOK, N.; YIM, M. Emulating self-reconfigurable robots - design of the SMORES system. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012. p. 4464–4469. ISBN 978-1-4673-1736-8. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6385845http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6385845>. Cited 2 times on pages vi e 9.

Digi International Inc. *XBee /XBee-PRO RF Modules Datasheet*. [S.l.], 2009. Cited on page 40.

ERİŞMİŞ, M. A. *MEMS Accelerometers and Gyroscopes for Inertial Measurements Units*. Tese (PhD) — Middle East Techincal University, 2004. Cited 2 times on pages 17 e 19.

FUKUDA, T.; KAWAUCHI, Y. Cellular robotic system (CEBOT) as one of the realization of self-organizing intelligent universal manipulator. In: *Proceedings., IEEE International Conference on Robotics and Automation*. Cincinnati, OH: IEEE Comput. Soc. Press, 1990. p. 662–667. ISBN 0-8186-9061-5. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=126059>. Cited on page 8.

GLASSER, P. C. An Introduction to the Use of Complementary Filters for Fusion of Sensor Data. Disponível em: <http://glassercommunications.com/paul/samples/filters_for_fusion.pdf>. Cited 2 times on pages vi e 23.

GONCALVES, L. L. C. et al. Kinematics of a Hexapod Modular Robot. In: *22nd International Congress of Mechanical Engineering (COBEM 2013)*. Ribeirão Preto, SP, Brazil: ABCM, 2013. Cited on page 65.

GREEN, K.; FURCHTGOTT-ROTH, D. Intermodal Safety in the Transport of Oil. *Studies in Energy Transportation*, n. October, p. 32, 2013. Disponível em: <http://ssrn.com/abstract=2345409>. Cited 2 times on pages 4 e 5.

HIGGINS, W. A Comparison of Complementary and Kalman Filtering. *IEEE Transactions on Aerospace and Electronic Systems*, AES-11, n. 3, p. 321–325, maio 1975. ISSN 0018-9251. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4101411>. Cited 4 times on pages vi, 22, 23 e 24.

HIROSE, S. et al. Design of in-pipe inspection vehicles for ϕ25, ϕ50, ϕ150 pipes. In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*. IEEE, 1999. v. 3, p. 2309–2314. ISBN 0-7803-5180-0. ISSN 1050-4729. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=770450>. Cited on page 5.

HORODINCA, M. et al. A simple architecture for in-pipe inspection robots. In: *International Colloquium on Mobile and Autonomous Systems*. Magdeburg, Germany: [s.n.], 2002. p. 061–064. Cited on page 6.

INUKTUM. *Crawler Vehicles*. Disponível em: <http://www.inuktun.com/crawler-vehicles/>. Cited on page 6.

INVENSENSE. *ITG-3200 Product Specification Datasheet*. [S.l.], 2010. v. 1, n. 408, 1–39 p. Cited on page 39.

JENSEN, C. *Numerical Simulation of Gyroscope Effects in Ansys*. 1–78 p. Tese (PhD Thesis) — Aalborg University, 2011. Cited 3 times on pages vi, 17 e 18.

JORGENSEN, M.; OSTERGAARD, E.; LUND, H. Modular ATRON: modules for a self-reconfigurable robot. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and*

Systems (IROS) (IEEE Cat. No.04CH37566). IEEE, 2004. v. 2, p. 2068–2073. ISBN 0-7803-8463-6. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1389702http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1389702>. Cited on page 8.

KOVVALI, N.; BANAVAR, M.; SPANIAS, A. An Introduction to Kalman Filtering with MATLAB Examples. *Synthesis Lectures on Signal Processing*, v. 6, n. 2, p. 1–81, set. 2013. ISSN 1932-1236. Disponível em: <http://www.morganclaypool.com/doi/abs/10.2200/S00534ED1V01Y201309SPR012>. Cited on page 22.

LEE, I. et al. Development and analysis of the vertical capacitive accelerometer. *Sensors and Actuators A: Physical*, v. 119, n. 1, p. 8–18, mar. 2005. ISSN 09244247. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0924424704004716>. Cited on page 17.

LIEDKE, J. et al. The Collective Self-reconfigurable Modular Organism (CoSMO). In: *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. IEEE, 2013. p. 1–6. ISBN 978-1-4673-5320-5. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6584059>. Cited 2 times on pages vi e 11.

MAGNUSSEN, O.; OTTESTAD, M.; HOVLAND, G. Experimental validation of a quaternion-based attitude estimation with direct input to a quadcopter control system. In: *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2013. p. 480–485. ISBN 978-1-4799-0817-2. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6564723>. Cited 4 times on pages vii, 30, 33 e 34.

MOECKEL, R. et al. Gait optimization for roombots modular robots — Matching simulation and reality. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013. p. 3265–3272. ISBN 978-1-4673-6358-7. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6696820http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6696820>. Cited 3 times on pages vi, 11 e 12.

MORRIS, A. S. *Measurement and Instrumentation Principles*. 3rd editio. ed. Oxford: Butterworth, 2001. ISSN 0957-0233. Cited 8 times on pages vi, 15, 16, 17, 19, 20, 21 e 22.

MURATA, S.; KUROKAWA, H. Self-reconfigurable robots. *IEEE Robotics & Automation Magazine*, v. 14, n. 1, p. 71–78, mar. 2007. ISSN 1070-9932. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4141035http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4141035>. Cited 3 times on pages vi, 7 e 8.

MURATA, S. et al. M-TRAN: self-reconfigurable modular robotic system. *IEEE/ASME Transactions on Mechatronics*, v. 7, n. 4, p. 431–441, dez. 2002. ISSN 1083-4435. Disponível em: <http://staff.aist.go.jp/e.yoshida/papers/dars2002.pdfhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1159221>. Cited on page 8.

NORBERG, A. Design Rules, Volume 1: The Power of Modularity [Book Review]. *IEEE Annals of the History of Computing*, MIT Press, Cambridge, MA, v. 23, n. 1, p. 65–66, jan. 2001. ISSN 1058-6180. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4496938>. Cited on page 6.

OHNO, H.; HIROSE, S. Design of slim slime robot and its gait of locomotion. *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, v. 2, 2001. Cited on page 6.

OKAMOTO, J. et al. Autonomous system for oil pipelines inspection. *Mechatronics*, v. 9, n. 7, p. 731–743, out. 1999. ISSN 09574158. Disponível em: <http://www.sciencedirect.com/science/

article/pii/S0957415899000318http://linkinghub.elsevier.com/retrieve/pii/S0957415899000318>. Cited on page 5.

PEDLEY, M. Tilt Sensing Using a Three-Axis Accelerometer. *Freescale Semiconductor Application Note*, p. 2012–2013, 2013. Cited on page 32.

PFEIFFER, F.; ROSSMANN, T.; LOFFLER, K. Control of a tube crawling machine. In: *2000 2nd International Conference. Control of Oscillations and Chaos. Proceedings (Cat. No.00TH8521)*. IEEE, 2000. v. 3, p. 586–591. ISBN 0-7803-6434-1. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=874339>. Cited on page 6.

QIAO, G. et al. Design of a self-reconfigurable wireless network system for modular self-reconfigurable robots. In: *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2012. p. 1337–1342. ISBN 978-1-4673-2127-3. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6491154>. Cited 3 times on pages vi, 12 e 13.

QIAO, G. et al. Design of transmote: A modular self-reconfigurable robot with versatile transformation capabilities. In: *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2012. p. 1331–1336. ISBN 978-1-4673-2127-3. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6491153>. Cited 3 times on pages vi, 12 e 13.

SETCHI, R.; LAGOS, N. Reconfigurability and reconfigurable manufacturing systems state-of-the-art review. In: *2nd IEEE International Conference on Industrial Informatics, 2004. INDIN '04. 2004.* IEEE, 2004. p. 529–535. ISBN 0-7803-8513-6. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1417401>. Cited on page 6.

SHARP. *GP2Y0A41SK0F Datasheet*. [S.l.], 2002. 1–9 p. Cited 5 times on pages xx, xxiii, 39, 54 e 62.

SOUSA, A. C. C. de et al. ErekoBot Sigma: Protótipo de um robô modular reconfigurável com sensores. In: *VIII Congresso Nacional de Engenharia Mecânica*. Uberlândia, MG, Brazil: [s.n.], 2014. Cited on page 65.

SOUSA, A. C. C. de; VIANA, D. M. a.; KOIKE, C. M. C. e. C. Data Communication and Protocols in self-reconfigurable modular robots. In: *22nd International Congress of Mechanical Engineering (COBEM 2013)*. Ribeirão Preto, SP, Brazil: ABCM, 2013. Cited 2 times on pages 13 e 65.

SOUSA, A. C. C. de; VIANA, D. M. a.; KOIKE, C. M. C. e. C. Programando a movimentação em robôs modulares autorreconfiguráveis ápodes para inspeção em tubulações. In: *7º PDPetro Congresso Brasileiro de P&D em Petróleo e Gás*. Aracaju, SE, Brazil: [s.n.], 2013. Cited on page 65.

SOUSA, A. C. C. de; VIANA, D. M. a.; KOIKE, C. M. C. e. C. Sensors in Reconfigurable Modular Robot for Pipeline Inspection : Design and Tests of a Prototype. In: *Joint Conference on Robotics and Intelligent Systems 2014*. São Carlos, SP, Brazil: [s.n.], 2014. Cited on page 65.

SPRöWITZ, a. et al. Roombots: A hardware perspective on 3D self-reconfiguration and locomotion with a homogeneous modular robot. *Robotics and Autonomous Systems*, Elsevier B.V., v. 62, n. 7, p. 1016–1033, jul. 2014. ISSN 09218890. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0921889013001632>. Cited 3 times on pages vi, 11 e 12.

STREICH, H.; ADRIA, O. Software approach for the autonomous inspection robot MAKRO. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004.* IEEE, 2004. v. 4, p. 3411–3416 Vol.4. ISBN 0-7803-8232-3. ISSN 1050-4729. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1308781>. Cited on page 6.

Torres Jr, C. R.; MANZAK, P. T.; MILLER, J. E. *Variable speed PIG for pipeline applications.* 2002. Cited on page 6.

USTA, U. Y. *Comparison of quaternion and Euler angle methods for joint angle animation of human figure models.* Tese (Master of Science in Computer Science) — Naval Postgraduate School, 1999. Cited 2 times on pages vi e 24.

VESPIGNANI, M. et al. An experimental study on the role of compliant elements on the locomotion of the self-reconfigurable modular robots Roombots. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems.* IEEE, 2013. p. 4308–4313. ISBN 978-1-4673-6358-7. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6696974>. Cited 3 times on pages vi, 11 e 12.

WANG, X.; ZHU, Y.; ZHAO, J. A dynamic simulation and virtual evolution platform for modular self-reconfigurable robots. In: *2013 IEEE International Conference on Information and Automation (ICIA).* IEEE, 2013. p. 457–462. ISBN 978-1-4799-1334-3. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6720342>. Cited 2 times on pages vi e 10.

YE, S. et al. Robust robotic grasping using IR Net-Structure Proximity Sensor to handle objects with unknown position and attitude. In: *2013 IEEE International Conference on Robotics and Automation.* IEEE, 2013. p. 3271–3278. ISBN 978-1-4673-5643-5. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6631033http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6564723>. Cited 2 times on pages vii e 35.

YIM, M. *Polypod: Locomotion With a Unit Modular Reconfigurable Robot.* Tese (PhD) — Stanford University, 1994. Cited 3 times on pages xv, 1 e 6.

YIM, M. et al. Connecting and disconnecting for chain self-reconfiguration with PolyBot. *IEEE/ASME Transactions on Mechatronics*, v. 7, n. 4, p. 442–451, dez. 2002. ISSN 1083-4435. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1159222>. Cited 2 times on pages 8 e 9.

ZHU, Y. et al. Design and implementation of UBot: A modular Self-Reconfigurable Robot. In: *2013 IEEE International Conference on Mechatronics and Automation.* IEEE, 2013. p. 1217–1222. ISBN 978-1-4673-5560-5. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6618087>. Cited 2 times on pages vi e 10.

Appendix

I. DEMONSTRATIONS

I.1 Six possible rotation matrix

$$\begin{aligned}
\vec{R}_{xyz}\begin{bmatrix}0\\0\\1\end{bmatrix} &= \vec{R}_x(\phi)\vec{R}_y(\theta)\vec{R}_z(\psi)\begin{bmatrix}0\\0\\1\end{bmatrix} \\
&= \begin{bmatrix}\cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta\\ \cos\psi\sin\theta\sin\phi-\cos\phi\sin\psi & \cos\theta\cos\psi+\sin\theta\sin\phi\sin\psi & \cos\theta\sin\phi\\ \cos\phi\cos\psi\sin\theta+\sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi-\cos\psi\sin\phi & \cos\theta\cos\phi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix} \\
&= \begin{bmatrix}-\sin\theta\\ \cos\theta\sin\phi\\ \cos\theta\cos\phi\end{bmatrix}
\end{aligned} \tag{I.1}$$

$$\begin{aligned}
\vec{R}_{yxz}\begin{bmatrix}0\\0\\1\end{bmatrix} &= \vec{R}_y(\theta)\vec{R}_x(\phi)\vec{R}_z(\psi)\begin{bmatrix}0\\0\\1\end{bmatrix} \\
&= \begin{bmatrix}\cos\psi\cos\theta-\sin\theta\sin\phi\sin\psi & \sin\psi\cos\theta+\sin\theta\sin\phi\cos\psi & -\sin\theta\cos\phi\\ -\cos\phi\sin\psi & \cos\theta\cos\psi & \sin\phi\\ \cos\theta\sin\phi\cos\psi+\sin\theta\cos\psi & -\cos\psi\cos\theta\sin\phi+\sin\psi\sin\theta & \cos\theta\cos\phi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix} \\
&= \begin{bmatrix}-\sin\theta\cos\phi\\ \sin\phi\\ \cos\theta\cos\phi\end{bmatrix}
\end{aligned} \tag{I.2}$$

$$\begin{aligned}
\vec{R}_{xzy}\begin{bmatrix}0\\0\\1\end{bmatrix} &= \vec{R}_x(\phi)\vec{R}_z(\psi)\vec{R}_y(\theta)\begin{bmatrix}0\\0\\1\end{bmatrix} \\
&= \begin{bmatrix}\cos\theta\cos\psi & \sin\psi & -\cos\psi\sin\theta\\ -\cos\phi\cos\theta\sin\psi+\sin\phi\sin\theta & \cos\phi\cos\psi & \cos\theta\sin\phi+\cos\phi\sin\theta\sin\psi\\ \cos\theta\sin\psi\sin\phi+\cos\phi\sin\theta & -\cos\psi\sin\theta & \cos\theta\cos\phi-\sin\theta\sin\phi\sin\psi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix} \\
&= \begin{bmatrix}-\cos\psi\sin\theta\\ \cos\theta\sin\phi+\cos\phi\sin\psi\sin\theta\\ \cos\phi\cos\theta-\sin\theta\sin\phi\sin\psi\end{bmatrix}
\end{aligned} \tag{I.3}$$

$$\vec{R}_{yzx}\begin{bmatrix}0\\0\\1\end{bmatrix} = \vec{R}_y(\theta)\vec{R}_z(\psi)\vec{R}_x(\phi)\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$= \begin{bmatrix}\cos\psi\cos\theta & \cos\phi\cos\theta\sin\psi+\sin\theta\sin\phi & \cos\theta\sin\phi\sin\psi-\sin\theta\cos\phi\\ -\sin\psi & \cos\phi\cos\psi & \cos\psi\sin\phi\\ \cos\psi\sin\theta & -\cos\theta\sin\phi+\cos\phi\sin\psi\sin\theta & \cos\theta\cos\phi+\sin\theta\sin\phi\sin\psi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$= \begin{bmatrix}\cos\theta\sin\phi\sin\psi-\cos\phi\sin\theta\\ \cos\psi\sin\phi\\ \cos\theta\cos\phi+\sin\theta\sin\phi\sin\psi\end{bmatrix} \tag{I.4}$$

$$\vec{R}_{zxy}\begin{bmatrix}0\\0\\1\end{bmatrix} = \vec{R}_z(\psi)\vec{R}_x(\phi)\vec{R}_y(\theta)\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$= \begin{bmatrix}\cos\psi\cos\theta+\sin\theta\sin\phi\sin\psi & \cos\phi\sin\psi & \cos\theta\sin\phi\sin\psi-\sin\theta\cos\psi\\ -\cos\theta\sin\psi+\cos\psi\sin\phi\sin\theta & \cos\phi\cos\psi & \cos\psi\cos\theta\sin\phi+\sin\theta\sin\psi\\ \cos\phi\sin\theta & -\sin\theta & \cos\theta\cos\phi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$= \begin{bmatrix}\cos\theta\sin\phi\sin\psi-\cos\psi\sin\theta\\ \cos\psi\cos\theta\sin\phi+\sin\theta\sin\psi\\ \cos\theta\cos\phi\end{bmatrix} \tag{I.5}$$

$$\vec{R}_{zyx}\begin{bmatrix}0\\0\\1\end{bmatrix} = \vec{R}_z(\psi)\vec{R}_y(\theta)\vec{R}_x(\phi)\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$= \begin{bmatrix}\cos\psi\cos\theta & \cos\phi\sin\psi+\cos\psi\sin\phi\sin\theta & \sin\phi\sin\psi-\cos\phi\cos\psi\sin\theta\\ -\cos\theta\sin\psi & \cos\psi\cos\phi+\sin\theta\sin\phi\sin\psi & \cos\psi\sin\phi+\cos\phi\sin\psi\sin\theta\\ \sin\theta & -\cos\theta\sin\phi & \cos\theta\cos\phi\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}$$
$$= \begin{bmatrix}\sin\phi\sin\psi-\cos\phi\cos\psi\sin\theta\\ \cos\psi\sin\phi+\cos\phi\sin\psi\sin\theta\\ \cos\theta\cos\phi\end{bmatrix} \tag{I.6}$$

II. LIBRARIES

We programed some libraries in C Language for the *ErekoBot σ* , and the architecture of the software works as follows (Figure II.1):

- The main.c controls the servo motor thought PWM signal using servo.h library.
- The main.c controls the LEDs connected to I/O pins using leds.h library.
- The main.c sends and receives messages through USART using USART.h library.
- The main.c receives data from the IR Sensors connected to AD ports using adc.h library.
  - The SHARP-0A41SKF36.h library converts the binary data to a volts metric system.
- The main.c receives data from the IMU connected to the I2C pins using i2c.h library.
  - The ADXL345.h holds the accelerometer address and saves the binary data to more readable variables.
  - The ITG-3200.h holds the gyroscope address and saves the binary data to more readable variables.

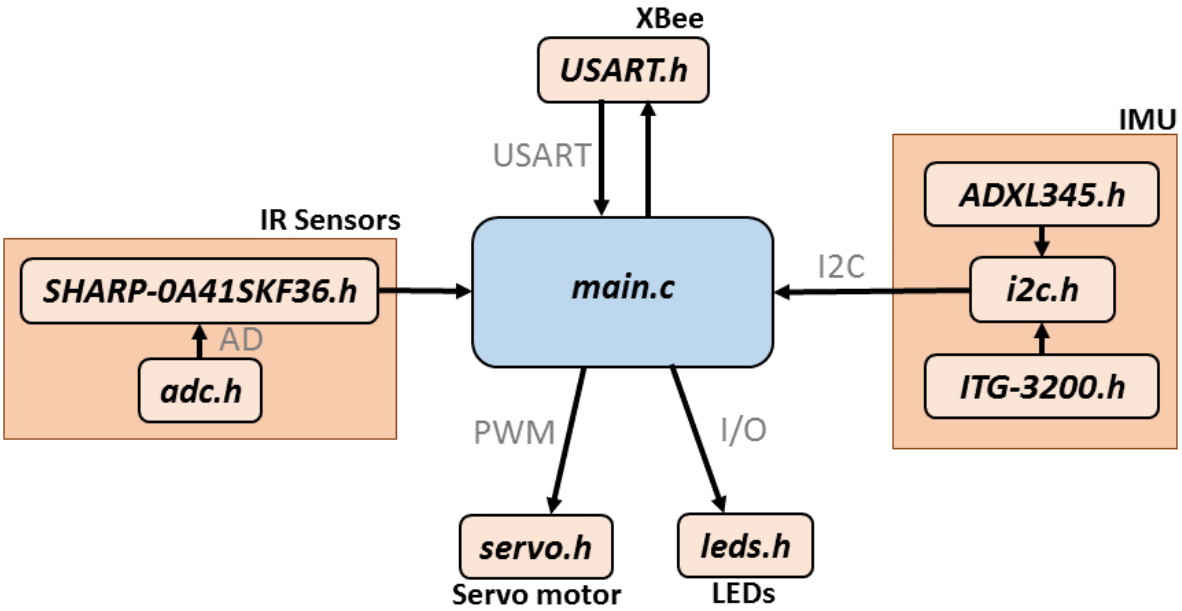


Figure II.1: Libraries architecture.

The *main.c* program depends of the routine of the experiments showed in Section 5.

II.1 Codes

We programed all codes n C Language and compiled them with avr-gcc in Ubuntu 12.04 LTS.

II.1.1 servo.h

```
#ifndef servo_HEADER_GUARD
#define servo_HEADER_GUARD

extern void servoInit(void);
extern void servoAngle(uint16_t angle);

#endif
```

Appendix–B/codes/servo.h

II.1.1.1 servo.c

```
/*
Setup servo motor.

@author Ana Carolina Cardoso de Sousa
@version 1.0

*/

#include <avr/io.h>
#include "servo.h"

void servoInit(void){

  TCCR1A |= (1<<COM1A1) | (1<<WGM11); // non-inverting mode for OC1A
  TCCR1B |= (1<<WGM13) | (1<<WGM12) | (1<<CS11); // Mode 14, Prescaler 8
  ICR1 = 40000; // 320000 / 8 = 40000
  DDRB |= (1<<PB1); // OC1A set to output

  servoAngle(3000);
}

void servoAngle(uint16_t angle){

  OCR1A = angle;

}
```

Appendix–B/codes/servo.c

II.1.2 leds.h

```
#ifndef leds_HEADER_GUARD
#define leds_HEADER_GUARD

#define RED 0
#define GREEN 1

extern void ledsInit(void);
extern void ledON(unsigned char cor);
extern void ledOFF(unsigned char cor);
```

```
#endif
```

Appendix–B/codes/leds.h

II.1.2.1 leds.c

```
/*
Setup leds.

@author Ana Carolina Cardoso de Sousa
@version 1.0

*/

#include <avr/io.h>
#include "leds.h"

void ledsInit(void){

  DDRB = _BV(PB0);        //Green Pin
  DDRD = _BV(PD7);        //Red Pin
}

void ledON(unsigned char cor){

  /*Switch for: receptor, transmissor, receptor and transmissor*/
  switch(cor){
    case GREEN:
      PORTB |= _BV(PB0); break;
    case RED:
      PORTD |= _BV(PD7); break;
    default:
      break;
  }
}

void ledOFF(unsigned char cor){

  /*Switch for: receptor, transmissor, receptor and transmissor*/
  switch(cor){
    case GREEN:
      PORTB &= ~_BV(PB0); break;
    case RED:
      PORTD &= ~_BV(PD7); break;
    default:
      break;
  }
}
```

Appendix–B/codes/leds.c

II.1.3 USART.h

```
#ifndef USART_HEADER_GUARD
#define USART_HEADER_GUARD
```

```
#define SYNC 0x44
#define RADDR 0x00
#define RECEPTOR 0
#define TRANSMISSOR 1
#define RECEPTOR_E_TRANSMISSOR 2

extern void USART_Init(unsigned int ubrr, unsigned char tipo);
extern void USART_TransmitByte(uint8_t data);
extern void USART_TransmitPackage(uint8_t sync, uint8_t addr, unsigned int cmd);
extern void USART_SensorPackage(uint8_t sync, uint8_t addr, unsigned int cmd);
extern void USART_TransmitSensorPackage(uint8_t sync, uint8_t addr, signed short int *gyro,
     signed short int *acc, uint16_t d1, uint16_t d2);

#endif
```

Appendix–B/codes/USART.h

II.1.3.1 USART.c

```
/* Setup USART.

@author Ana Carolina Cardoso de Sousa
@version 2.0*/

#include <avr/io.h>
#include "USART.h"

/*Setup frame: 8-bit, 2 stop bits. Setup boudrate: ubrr.

@param ubrr boudrate
@param tipo type of communication, receptor, transmissor, receptor and transmissor.*/

void USART_Init(unsigned int ubrr, unsigned char tipo){

  UBRR0H = (unsigned char)(ubrr >> 8);  /*baudrate high*/
  UBRR0L = (unsigned char) ubrr;    /*baudrate low*/

  /*Frame: asynchronous, no parity, 8-bit, 1 stop bit*/
  UCSR0C |= (1<<UMSEL01) | (0<<UMSEL00) | (0<<UPM01) | (0<<UPM00) | (0<<USBS0) | (1<<UCSZ01
    ) | (1<<UCSZ00) | (0<<UCPOL0);

  /*Switch for: receptor, transmissor, receptor and transmissor*/
  switch(tipo){
    case RECEPTOR:
      UCSR0B = (1 << RXEN0) | (1 << RXCIE0); break;
    case TRANSMISSOR:
      UCSR0B = (1 << TXEN0); break;
    default:
      UCSR0B = (1 << RXEN0) | (1 << RXCIE0) | (1 << TXEN0); break;
  }

}

/*Write in the buffer.

@param data Byte transmited.*/

```

```
void USART_TransmitByte(unsigned char data){

  while( (UCSR0A&(1<<UDRE0)) == 0 );     /*Wait the transmission buffer clear*/

  UDR0 = data;           /*Put it at the buffer (send byte)*/
}

/*Write data the buffer.

@param addr SYNC Byte
@param addr Receptor address
@param cmd  Data*/

void USART_TransmitPackage(uint8_t sync, uint8_t addr, unsigned int cmd){

  uint8_t chk;
  chk = addr+cmd;

  /*Send Package*/
  USART_TransmitByte(sync);      //sync byte
  USART_TransmitByte(addr);      //receptor address byte
  USART_TransmitByte((unsigned char)(cmd >> 8));  //data high 8 bits
  USART_TransmitByte((unsigned char)cmd);    //data low 8 bits
  USART_TransmitByte(chk);       //checksum
}

/*Transmit sensor package

@param addr Sync Byte
@param addr Receptor address
@param gyro Gyro Data
@param acc  Acc Data

*/

void USART_TransmitSensorPackage(uint8_t sync, uint8_t addr, signed short int *gyro, signed
     short int *acc, uint16_t d1, uint16_t d2){

  uint8_t chk;
  signed short int gx,gy,gz;
  signed short int ax,ay,az;
  chk = addr+sync;

  //Get gyro
  gx = gyro[0];
  gy = gyro[1];
  gz = gyro[2];

  //Get acc
  ax = acc[0];
  ay = acc[1];
  az = acc[2];

  /*Send Package*/
  USART_TransmitByte(sync);        //sync byte
  USART_TransmitByte(addr);        //receptor address byte

  USART_TransmitByte((unsigned char)(gyro[0] >> 8));  //data high 8 bits gyro
  USART_TransmitByte((unsigned char)gyro[0]);    //data low 8 bits gyro
  USART_TransmitByte((unsigned char)(gyro[1] >> 8));  //data high 8 bits gyro
```

```
97    USART_TransmitByte((unsigned char)gyro[1]);    //data low 8 bits gyro
      USART_TransmitByte((unsigned char)(gyro[2] >> 8));  //data high 8 bits gyro
99    USART_TransmitByte((unsigned char)gyro[2]);    //data low 8 bits gyro

101   USART_TransmitByte((unsigned char)(acc[0] >> 8)); //data high 8 bits acc
      USART_TransmitByte((unsigned char)acc[0]);     //data low 8 bits acc
103   USART_TransmitByte((unsigned char)(acc[1] >> 8)); //data high 8 bits acc
      USART_TransmitByte((unsigned char)acc[1]);     //data low 8 bits acc
105   USART_TransmitByte((unsigned char)(acc[2] >> 8)); //data high 8 bits acc
      USART_TransmitByte((unsigned char)acc[2]);     //data low 8 bits acc
107
      USART_TransmitByte((unsigned char)(d1 >> 8)); //data high 8 bits IR1
109   USART_TransmitByte((unsigned char)d1);     //data low 8 bits IR1
      USART_TransmitByte((unsigned char)(d2 >> 8)); //data high 8 bits IR2
111   USART_TransmitByte((unsigned char)d2);     //data low 8 bits IR2

113   USART_TransmitByte(chk);         //checksum
}
```

Appendix–B/codes/USART.c

II.1.4 adc.h

```
1  #ifndef adc_HEADER_GUARD
   #define adc_HEADER_GUARD
3
   extern void adcInit(void);
5  extern int16_t adcRead(char ch);
   extern int16_t adcReadMean(char ch);
7
   #endif
```

Appendix–B/codes/adc.h

II.1.4.1 adc.c

```
   /*
2  ADC functions.

4  @author Ana Carolina Cardoso de Sousa
   @date 2014/05/12
6  @version 1.0
   */
8
   #include <avr/io.h>
10 #include <stdint.h>
   #include "adc.h"
12
   /* Initializes the ADC. */
14 void adcInit(void){

16   //AREF = AVcc, ADC left adjust result
     ADMUX = (0<<REFS1)|(1<<REFS0)|(1<<ADLAR);
18   //ADC Enable, Prescaler of 128 (16M/128 = 125k)
     ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0);
```

```
}

/* Reads the value of the ADC. */
int16_t adcRead(char ch) {

  //Select the corresponding channel 0~7
  ch &= 0b00000111; //always keep the value of ch between 0 and 7
  ADMUX = (ADMUX & 0xF8)|ch; // clears the bottom 3 bits before ORing

  //Start single convertion
  ADCSRA |= (1<<ADSC);

  //Wait for conversion to complete
  while(ADCSRA & (1<<ADSC));

  return (ADC);
}

/* Reads many values of ADC. */
int16_t adcReadMean(char ch) {

  int32_t temp = 0;
  int16_t adc = 0;
  int8_t i = 0;

  for (i=0;i<5;i++){
    temp += adcRead(ch);
  }

  adc = temp/5;

  return (adc);
}
```

Appendix–B/codes/adc.c

II.1.5 SHARP-0A41SKF36.h

```
#ifndef SHARP0A41SKF36_HEADER_GUARD
#define SHARP0A41SKF36_HEADER_GUARD

#define T1 00
#define T2 01
#define F1 02
#define F2 03

extern int16_t shift6(int16_t adc);
extern int16_t adc2volts(int32_t adc);
extern int16_t *readFront(void);
extern int16_t *readBack(void);

#endif
```

Appendix–B/codes/SHARP–0A41SKF36.h

II.1.5.1 SHARP-0A41SKF36.c

```
/*
SHARP-0A41SKF36 functions.

@author Ana Carolina Cardoso de Sousa
@date 2014/05/30
@version 1.0
*/

#include <stdint.h>
#include "SHARP-0A41SKF36.h"
#include "adc.h"

/*Shift adc value*/
int16_t shift6(int16_t adc){
  return adc = adc >> 6;
}

/*Converts adc to volts*/
int16_t  adc2volts(int32_t adc){
  int64_t x;

  x = shift6(adc);
  x = (5000*x); // 5*1000*x
  x = (x>>10);  // result = x/1024

  return (int16_t)x;

}

/* Reads the IR Sensors 'Front' values */
int16_t *readFront(void) {

  static int16_t adc[2];

  adc[0] = adcRead(F1);       // read adc value at F1
  adc[1] = adcRead(F2);       // read adc value at F2

  return adc;
}

/* Reads the IR Sensors 'Back' values */
int16_t *readBack(void) {

  static int16_t adc[2];

  adc[0] = adcRead(T1);       // read adc value at T1
  adc[1] = adcRead(T2);       // read adc value at T2

  return adc;
}
```

Appendix–B/codes/SHARP–0A41SKF36.c

II.1.6 i2c.h

```
#ifndef i2c_HEADER_GUARD
#define i2c_HEADER_GUARD
```

```
4 #include <avr/io.h>
  #include <avr/interrupt.h>
6 #include <util/delay.h>
  #include "types.h"
8 #include "defs.h"

10 #define TW_START         0x08
  #define TW_REP_START         0x10
12 #define TW_MT_SLA_ACK        0x18
  #define TW_MT_SLA_NACK         0x20
14 #define TW_MT_DATA_ACK         0x28
  #define TW_MT_DATA_NACK        0x30
16 #define TW_MT_ARB_LOST         0x38
  #define TW_MR_ARB_LOST         0x38
18 #define TW_MR_SLA_ACK        0x40
  #define TW_MR_SLA_NACK         0x48
20 #define TW_MR_DATA_ACK         0x50
  #define TW_MR_DATA_NACK        0x58
22 #define TW_ST_SLA_ACK        0xA8
  #define TW_ST_ARB_LOST_SLA_ACK       0xB0
24 #define TW_ST_DATA_ACK         0xB8
  #define TW_ST_DATA_NACK        0xC0
26 #define TW_ST_LAST_DATA        0xC8
  #define TW_SR_SLA_ACK        0x60
28 #define TW_SR_ARB_LOST_SLA_ACK       0x68
  #define TW_SR_GCALL_ACK        0x70
30 #define TW_SR_ARB_LOST_GCALL_ACK     0x78
  #define TW_SR_DATA_ACK         0x80
32 #define TW_SR_DATA_NACK        0x88
  #define TW_SR_GCALL_DATA_ACK       0x90
34 #define TW_SR_GCALL_DATA_NACK      0x98
  #define TW_SR_STOP         0xA0
36 #define TW_NO_INFO         0xF8
  #define TW_BUS_ERROR         0x00
38 #define TWCR_CMD_MASK        0x0F
  #define TWSR_STATUS_MASK       0xF8
40 #define I2C_OK           0x00
  #define I2C_ERROR_NODEV        0x01
42
  #define sbi(var, mask)   ((var) |= (uint8_t)(1 << mask))
44 #define cbi(var, mask)   ((var) &= (uint8_t)~(1 << mask))

46 #define WRITE_sda() DDRC = DDRC | 0b00010000 //SDA must be output when writing
  #define READ_sda()  DDRC = DDRC & 0b11101111 //SDA must be input when reading - don't
      forget the resistor on SDA!!
48
  // functions
50 void i2cInit(void);
  void i2cSendStart(void);
52 void i2cSendStop(void);
  void i2cWaitForComplete(void);
54 void i2cSendByte(unsigned char data);
  void i2cReceiveByte(unsigned char ackFlag);
56 unsigned char i2cGetReceivedByte(void);
  unsigned char i2cGetStatus(void);
58
  #endif
```

Appendix–B/codes/i2c.h

II.1.6.1 SHARP-0A41SKF36.c

```
/*
I2C functions

@author Ana Carolina Cardoso de Sousa
@date 2014/04/23
@version 1.0
*/

#include "i2c.h"

// Initialize I2C (TWI) interface
void i2cInit(void){
  cbi(TWSR, TWPS0);
  cbi(TWSR, TWPS1);

  outb(TWBR, 12);   // That just work for 16MHz, look at:
        // ATMega8 DataSheet table (CPU and SCL I2C frequency)
  sbi(TWCR, TWEN);  // Enable TWI
   _delay_ms(200);
}

// Low-level I2C transaction commands
// Send an I2C start condition in Master mode
void i2cSendStart(void){
  WRITE_sda();
  // send start condition
  TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);
}

// Send an I2C stop condition in Master mode
void i2cSendStop(void){
  // transmit stop condition
    TWCR = (1<<TWINT)|(1<<TWEN)|(1<<TWSTO);
}

// Wait for current I2C operation to complete
void i2cWaitForComplete(void){
  int i = 0;     //time out variable

  // wait for i2c interface to complete operation
    while ((!(TWCR & (1<<TWINT))) && (i < 90))
    i++;
}

// Send an (address|R/W) combination or a data byte over I2C
void i2cSendByte(unsigned char data){
  _delay_ms(1);
  //printf("sending 0x%x\n", data);
  WRITE_sda();
  // save data to the TWDR
  TWDR = data;
  // begin send
  TWCR = (1<<TWINT)|(1<<TWEN);
}

//! Receive a data byte over I2C
// ackFlag = TRUE if recevied data should be ACK'ed
// ackFlag = FALSE if recevied data should be NACK'ed
```

```
59 void i2cReceiveByte(unsigned char ackFlag){
     // begin receive over i2c
61   if( ackFlag ) {
       // ackFlag = TRUE: ACK the recevied data
63     outb(TWCR, (inb(TWCR)&TWCR_CMD_MASK)|BV(TWINT)|BV(TWEA));
     }
65   else  {
       // ackFlag = FALSE: NACK the recevied data
67     outb(TWCR, (inb(TWCR)&TWCR_CMD_MASK)|BV(TWINT));
     }
69 }

71 // Pick up the data that was received with i2cReceiveByte()
   unsigned char i2cGetReceivedByte(void){
73   // retieve received data byte from i2c TWDR
     return( inb(TWDR) );
75 }

77 // Get current I2c bus status from TWSR
   unsigned char i2cGetStatus(void){
79   // retieve current i2c status from i2c TWSR
     return( inb(TWSR) );
81 }
```

Appendix–B/codes/i2c.c

II.1.7 ADXL-345.h

```
1  #ifndef ADXL345_HEADER_GUARD
   #define ADXL345_HEADER_GUARD
3
   // ADXL-345 addresses
5  #define ADXL345_R 0xA7  // ADD pin is pulled high
   #define ADXL345_W 0xA6  // So address is 0x53
7
   // ADXL-345 registers
9  #define WHO    0x00
   #define BW_Rate    0x2C
11 #define PWR_CTL    0x2D
   #define INT_SRC    0x30
13 #define DATA_FORMAT 0x31
   #define AX_L     0x32
15 #define AX_H     0x33
   #define AY_L     0x34
17 #define AY_H     0x35
   #define AZ_L     0x36
19 #define AZ_H     0x37
   #define FIFO_CTL   0x38
21 #define FIFO_STATUS 0x39

23 // Macros
   #ifndef sbimacro_HEADER_GUARD
25 #define sbimacro_HEADER_GUARD
   #define sbi(var, mask)   ((var) |= (uint8_t)(1 << mask))
27 #define cbi(var, mask)   ((var) &= (uint8_t)~(1 << mask))
   #endif
29
   // ADXL-345 definitions
```

```
#define AX 1
#define AY 2
#define AZ 3

extern void ADXL345Init();
extern char ADXL345Read(unsigned char address);
extern void ADXL345Write(unsigned char address, unsigned char data);
extern signed short int * ADXL345get_a();

#endif
```

Appendix–B/codes/ADXL–345.h

II.1.7.1 SHARP-0A41SKF36.c

```
/*
ADXL-345 functions.

@author Ana Carolina Cardoso de Sousa
@date 2014/04/29
@version 1.0
*/
#include <avr/io.h>
#include <util/delay.h>
#include "ADXL-345.h"
#include "types.h"
#include "i2c.h"

void ADXL345Init(){
  ADXL345Write(PWR_CTL, 0x00);     // Go into standby mode to configure the device.
  ADXL345Write(DATA_FORMAT, 0x0B);  // 0x0B : Full resolution, +/-16g, 4mg/LSB.
  ADXL345Write(BW_Rate, 0x0D);     // 800Hz data rate.
  ADXL345Write(PWR_CTL, 0x08);     // 0x08 : Measurement mode.
  _delay_ms(22);
}

char ADXL345Read(unsigned char address){
  char data;

  cbi(TWCR, TWEN);   // Disable TWI
  sbi(TWCR, TWEN);   // Enable TWI

  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte(ADXL345_W); // write 0xA6
  i2cWaitForComplete();

  i2cSendByte(address); // write register address
  i2cWaitForComplete();

  i2cSendStart();

  i2cSendByte(ADXL345_R); // write 0xA7
  i2cWaitForComplete();
  i2cReceiveByte(FALSE);
  i2cWaitForComplete();

```

```
  data = i2cGetReceivedByte();  // Get MSB result
  i2cWaitForComplete();
  i2cSendStop();

  cbi(TWCR, TWEN);  // Disable TWI
  sbi(TWCR, TWEN);  // Enable TWI

  return data;
}

void ADXL345Write(unsigned char address, unsigned char data){
  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte(ADXL345_W);
  i2cWaitForComplete();

  i2cSendByte(address); // write register address
  i2cWaitForComplete();

  i2cSendByte(data);
  i2cWaitForComplete();

  i2cSendStop();
}

signed short int * ADXL345get_a(){

  signed short int ax,ay,az;
  static signed short int a_result[3];
  char temp;

  signed int accx=0,accy=0,accz=0;

  //Get accelerations:
  while (!(ADXL345Read(INT_SRC) & 0x01))  ;
  temp = 0;
  temp = ADXL345Read(AX_H); //high bits ax
  ax = temp << 8;
  ax = ADXL345Read(AX_L);   //low bits ax

  while (!(ADXL345Read(INT_SRC) & 0x01))  ;
  temp = 0;
  temp = ADXL345Read(AY_H); //high bits ay
  ay = temp << 8;
  ay |= ADXL345Read(AY_L);  //low bits ay

  while (!(ADXL345Read(INT_SRC) & 0x01))  ;
  temp = 0;
  temp = ADXL345Read(AZ_H); //high bits az
  az = temp << 8;
  az |= ADXL345Read(AZ_L);  //low bits az

  a_result[0] = ax;
  a_result[1] = ay;
  a_result[2] = az;

  return a_result;
}
```

Appendix–B/codes/ADXL–345.c

II.1.8 ITG-3200.h

```
#ifndef ITG3200_HEADER_GUARD
#define ITG3200_HEADER_GUARD

// ITG-3200 addresses
#define ITG3200_R 0xD1  // ADD pin is pulled high
#define ITG3200_W 0xD0  // So address is 0x68

// ITG-3200 registers
#define WHO 0x00
#define SMPL  0x15
#define DLPF  0x16
#define INT_C 0x17
#define INT_S 0x1A
#define TMP_H 0x1B
#define TMP_L 0x1C
#define GX_H  0x1D
#define GX_L  0x1E
#define GY_H  0x1F
#define GY_L  0x20
#define GZ_H  0x21
#define GZ_L  0x22
#define PWR_M 0x3E

// Macros
#ifndef sbimacro_HEADER_GUARD
#define sbimacro_HEADER_GUARD
#define sbi(var, mask)   ((var) |= (uint8_t)(1 << mask))
#define cbi(var, mask)   ((var) &= (uint8_t)~(1 << mask))
#endif

// ITG-3200 definitions
#define GYRO_SCALE 14.375
#define GX 1
#define GY 2
#define GZ 3

extern void ITG3200Init(void);
extern char ITG3200Read(unsigned char address);
extern void ITG3200Write(unsigned char address, unsigned char data);
extern int  ITG3200checkInterrupt(void);
extern signed short int * ITG3200calibration();
extern signed short int *ITG3200get_w(signed short int *gyro_offset);
//extern signed short int *ITG3200get_w_average(signed short int *gyro_offset);

#endif
```

Appendix–B/codes/ITG–3200.h

II.1.8.1 SHARP-0A41SKF36.c

```
/*
ITG-3200 functions.

@author Ana Carolina Cardoso de Sousa
@date 2014/04/28
@version 1.0
*/
#include <avr/io.h>
#include <util/delay.h>
#include "ITG-3200.h"
#include "types.h"
#include "i2c.h"

void ITG3200Init(void){
  ITG3200Write(PWR_M, 0x80);  // Reset to defaults
  ITG3200Write(SMPL, 0x00); // SMLPRT_DIV = 0
  ITG3200Write(DLPF, 0x18); // DLPF_CFG = 0, FS_SEL = 3
  ITG3200Write(INT_C, 0x05);  // Generate interrupt when device is ready or raw data ready
  ITG3200Write(PWR_M, 0x00);
  _delay_ms(50);
}

char ITG3200Read(unsigned char address){
  char data;

  cbi(TWCR, TWEN);  // Disable TWI
  sbi(TWCR, TWEN);  // Enable TWI

  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte(ITG3200_W); // write 0xD1
  i2cWaitForComplete();

  i2cSendByte(address); // write register address
  i2cWaitForComplete();

  i2cSendStart();

  i2cSendByte(ITG3200_R); // write 0xD3
  i2cWaitForComplete();
  i2cReceiveByte(FALSE);
  i2cWaitForComplete();

  data = i2cGetReceivedByte();  // Get MSB result
  i2cWaitForComplete();
  i2cSendStop();

  cbi(TWCR, TWEN);  // Disable TWI
  sbi(TWCR, TWEN);  // Enable TWI

  return data;
}

void ITG3200Write(unsigned char address, unsigned char data){
  i2cSendStart();
  i2cWaitForComplete();

  i2cSendByte(ITG3200_W); // write 0xB4
```

```
  i2cWaitForComplete();

  i2cSendByte(address); // write register address
  i2cWaitForComplete();

  i2cSendByte(data);
  i2cWaitForComplete();

  i2cSendStop();
}

/*Get offset for calibration*/
signed short int * ITG3200calibration(){
  char temp;
  signed int gx[10],gy[10],gz[10];
  signed int gyrox=0,gyroy=0,gyroz=0;
  unsigned int i;

  static short int gyro[3];


  for (i = 0; i<10; i++){
    while (!(ITG3200Read(INT_S) & 0x01))
    ;
    temp = 0;
    temp = ITG3200Read(GX_H);
    gx[i] = temp << 8;
    gx[i] |= ITG3200Read(GX_L);

    gyrox += gx[i];
  }

  for (i = 0; i<10; i++){
    while (!(ITG3200Read(INT_S) & 0x01))
    ;
    temp = 0;
    temp = ITG3200Read(GY_H);
    gy[i] = temp << 8;
    gy[i] |= ITG3200Read(GY_L);

    gyroy += gy[i];
  }

  for (i = 0; i<10; i++){
    while (!(ITG3200Read(INT_S) & 0x01))
    ;
    temp = 0;
    temp = ITG3200Read(GZ_H);
    gz[i] = temp << 8;
    gz[i] |= ITG3200Read(GZ_L);

    gyroz += gz[i];
  }

  gyrox = gyrox/10;
  gyroy = gyroy/10;
  gyroz = gyroz/10;

  gyrox = gyrox/GYRO_SCALE;
  gyroy = gyroy/GYRO_SCALE;
```

```
  gyroz = gyroz/GYRO_SCALE;

  gyro[0] = gyrox;
  gyro[1] = gyroy;
  gyro[2] = gyroz;

  return gyro;
}

/*Get angular velocities w=[wx,wy,wz]*/
signed short int *ITG3200get_w(signed short int *gyro_offset){

  signed short int gx,gy,gz;
  signed short int gx_off,gy_off,gz_off;
  static signed short int w_result[3];
  char temp;

  //Get offset
  gx_off = gyro_offset[0];
  gy_off = gyro_offset[1];
  gz_off = gyro_offset[2];

  //Get angular velocities:
  while (!(ITG3200Read(INT_S) & 0x01))  ;
  temp = 0;
  temp = ITG3200Read(GX_H);
  gx = temp << 8;      //high wx
  gx |= ITG3200Read(GX_L);  //high and low wx

  while (!(ITG3200Read(INT_S) & 0x01))  ;
  temp = 0;
  temp = ITG3200Read(GY_H);
  gy = temp << 8;      //high wy
  gy |= ITG3200Read(GY_L);  //high and low wy

  while (!(ITG3200Read(INT_S) & 0x01))  ;
  temp = 0;
  temp = ITG3200Read(GZ_H);
  gz = temp << 8;      //high wz
  gz |= ITG3200Read(GZ_L);  //high and low wz

  //Angular Velocity scaled without offset
  gx = gx/GYRO_SCALE;
  gy = gy/GYRO_SCALE;
  gz = gz/GYRO_SCALE;

  gx = (gx - gx_off);
  gy = (gy - gy_off);
  gz = (gz - gz_off);

  w_result[0] = gx;
  w_result[1] = gy;
  w_result[2] = gz;

  return w_result;
}
```

Appendix–B/codes/ITG–3200.c

III. ELECTRONIC CIRCUIT

We draw the both circuit schematics in CadSoft EAGLE PCB Design Software v6.5.0 (Figures III.1 and III.2). From these schemes, we generated the boards (Figures III.3, III.4, III.5 and fig:lateral-boardBW), printed and manually soldered the PCB components.

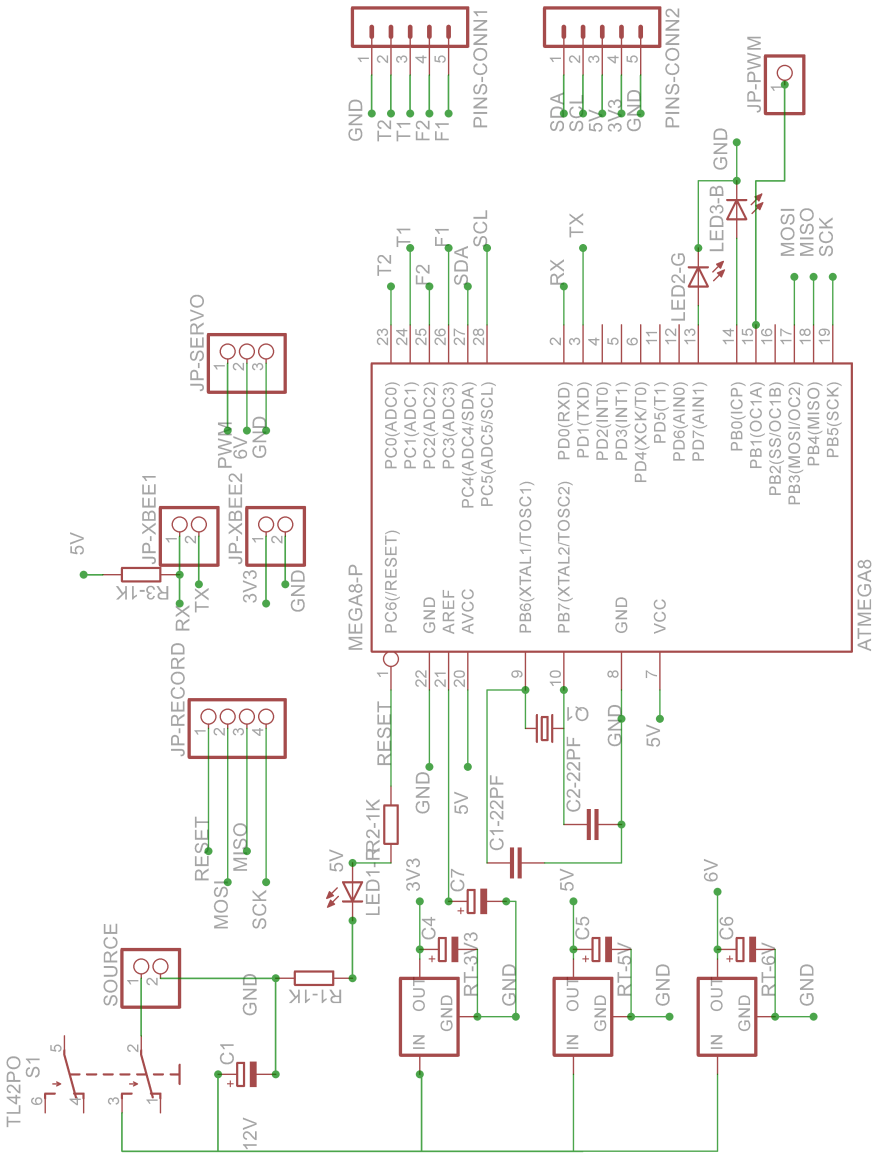


Figure III.1: Base Circuit Schematic.

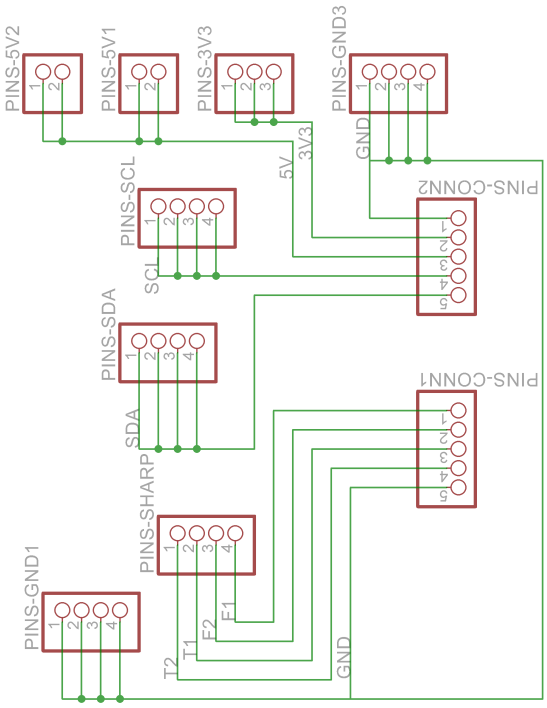


Figure III.2: Lateral Circuit Schematic.

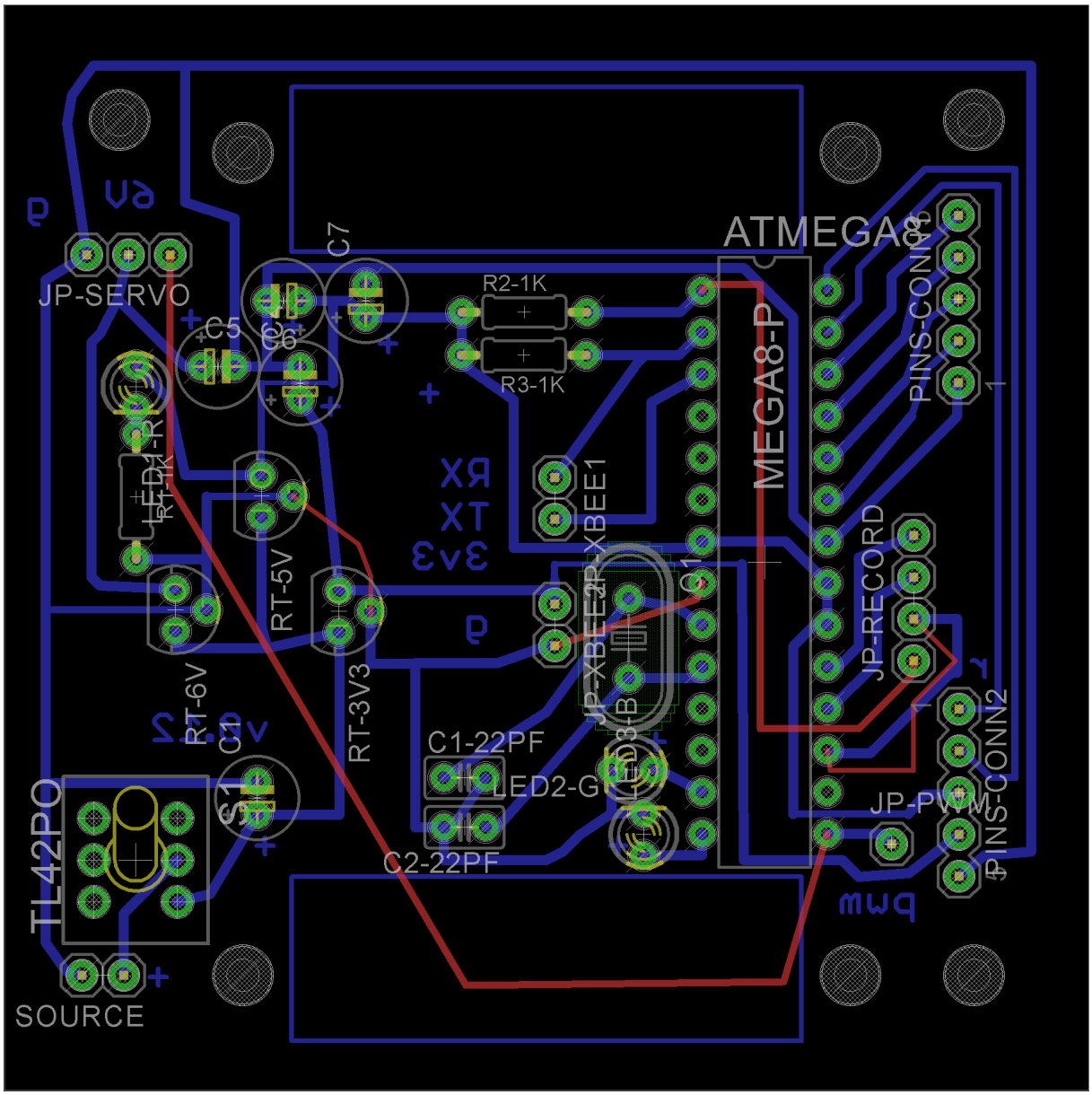


Figure III.3: Colored Base Circuit Board.

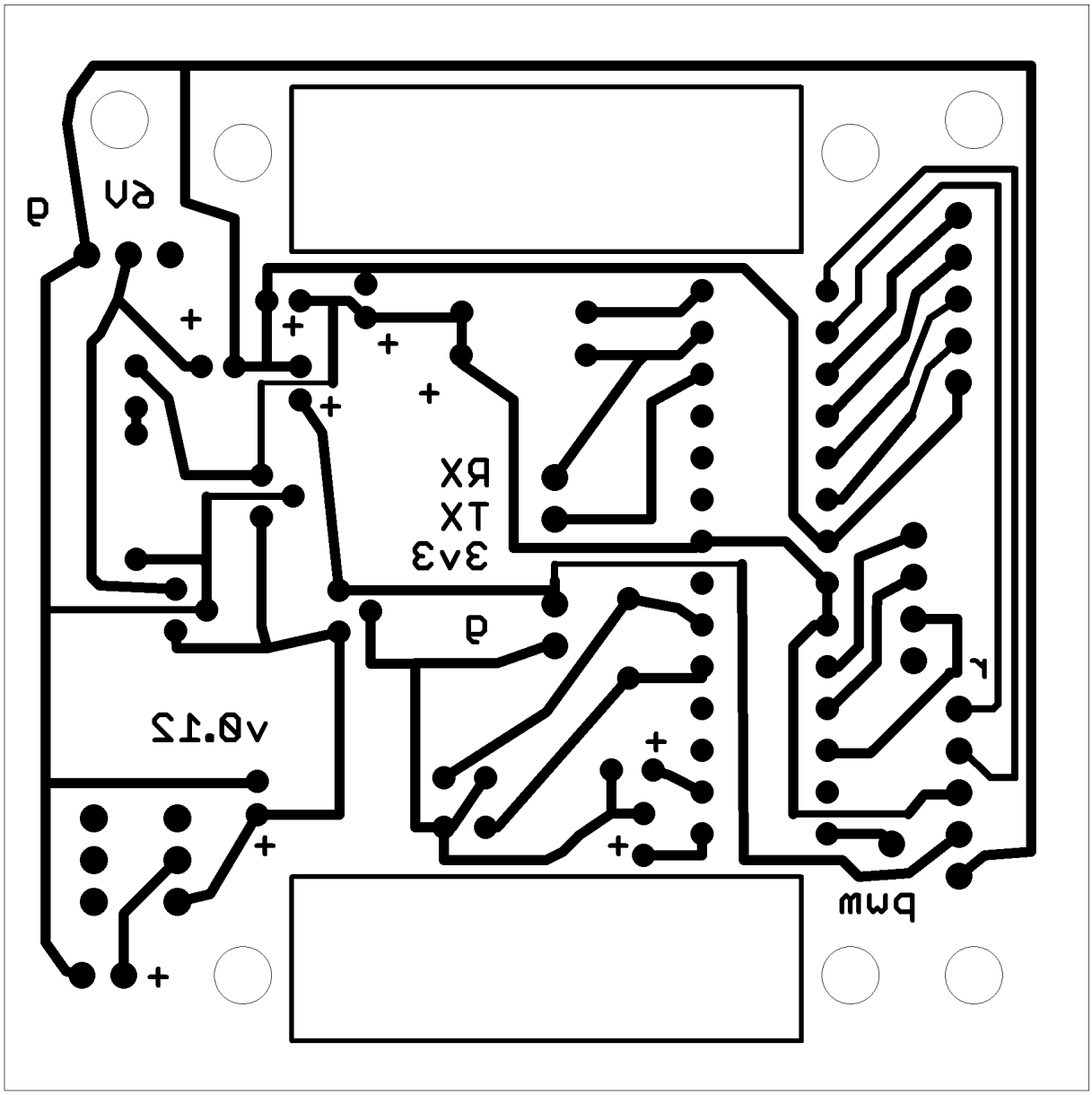


Figure III.4: Black and White Base Circuit Board.

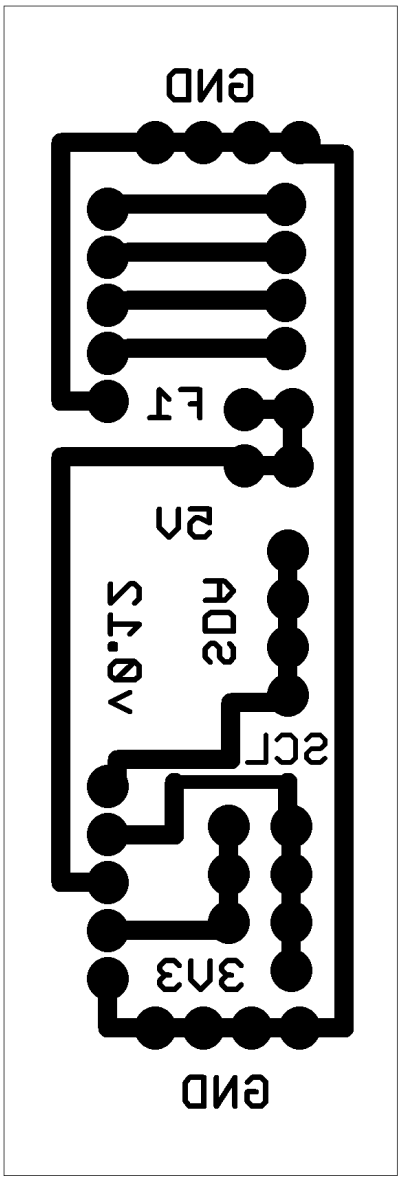


Figure III.5: Colored Lateral Circuit Board.

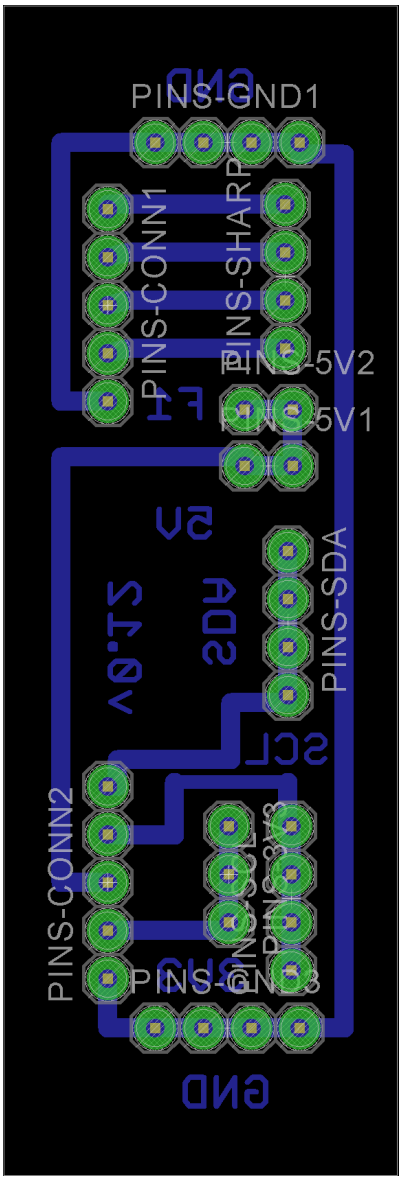


Figure III.6: Black and White Lateral Circuit Board.

IV. *ErekoBot* σ PLANS

We draw the all pieces in the CAD Software SolidWorks 2013, as we can see in Figures IV.1, IV.2, IV.3 and IV.4.

Figures IV.5, IV.6 and IV.7 show where to blend the pieces.

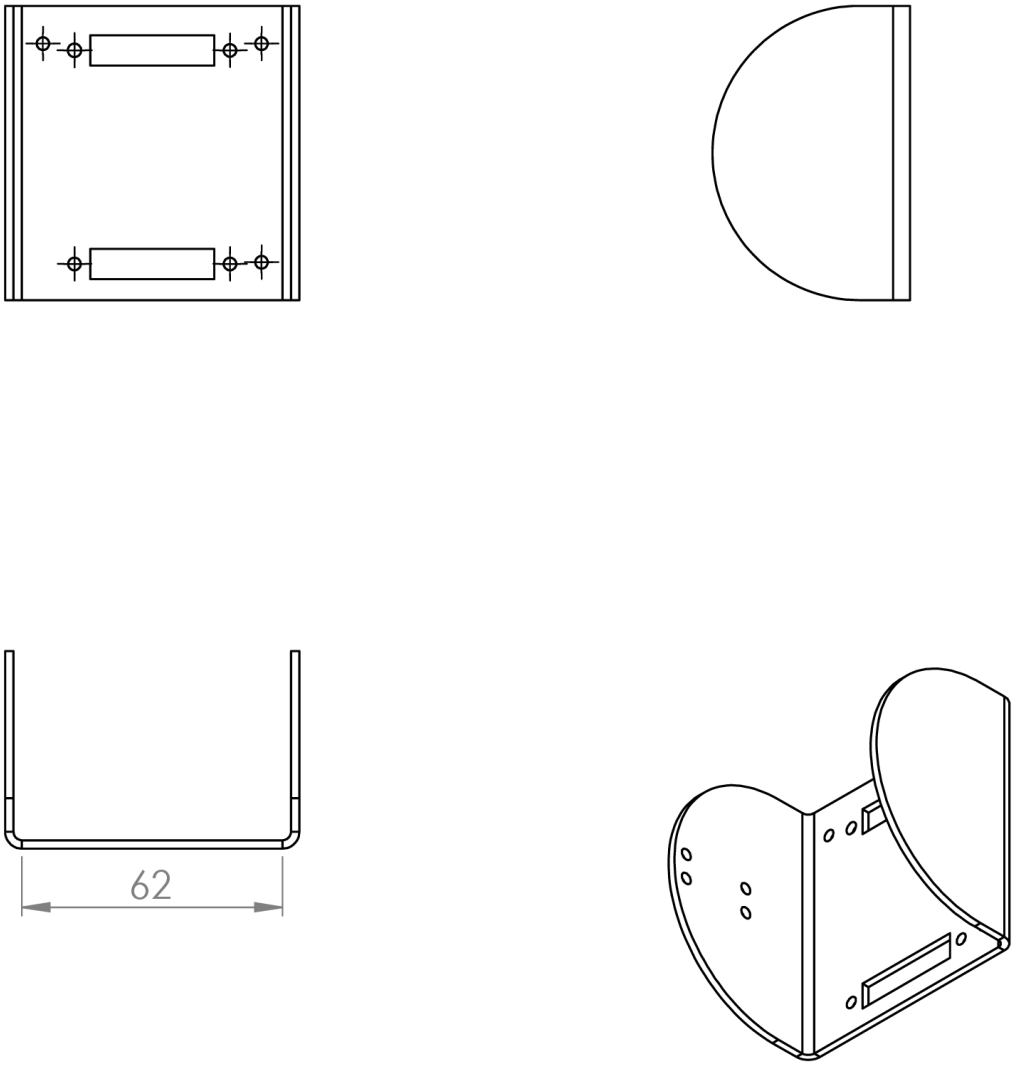


Figure IV.1: Base Plan.

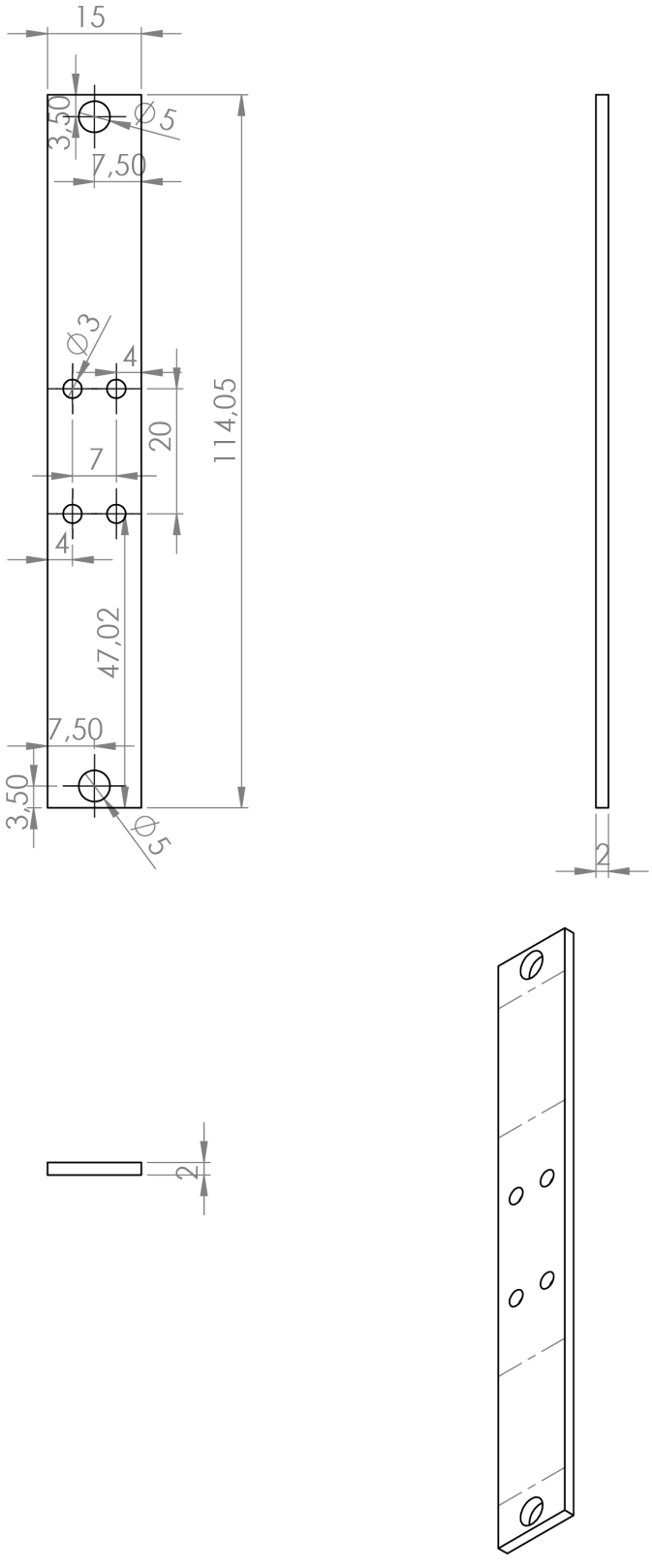


Figure IV.2: Motor Holder Plan.

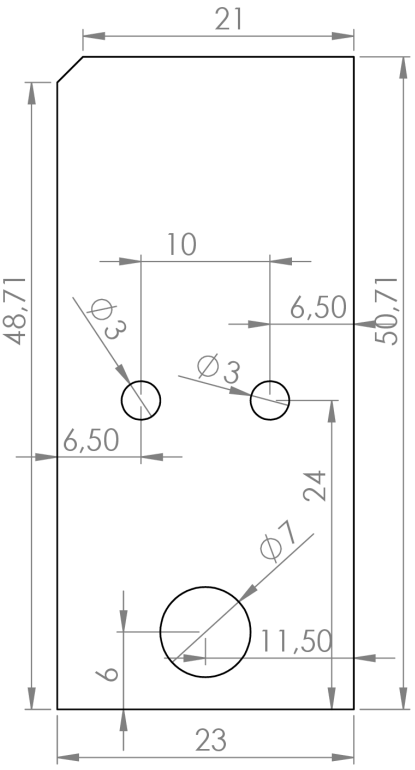


Figure IV.3: Rod Plan.

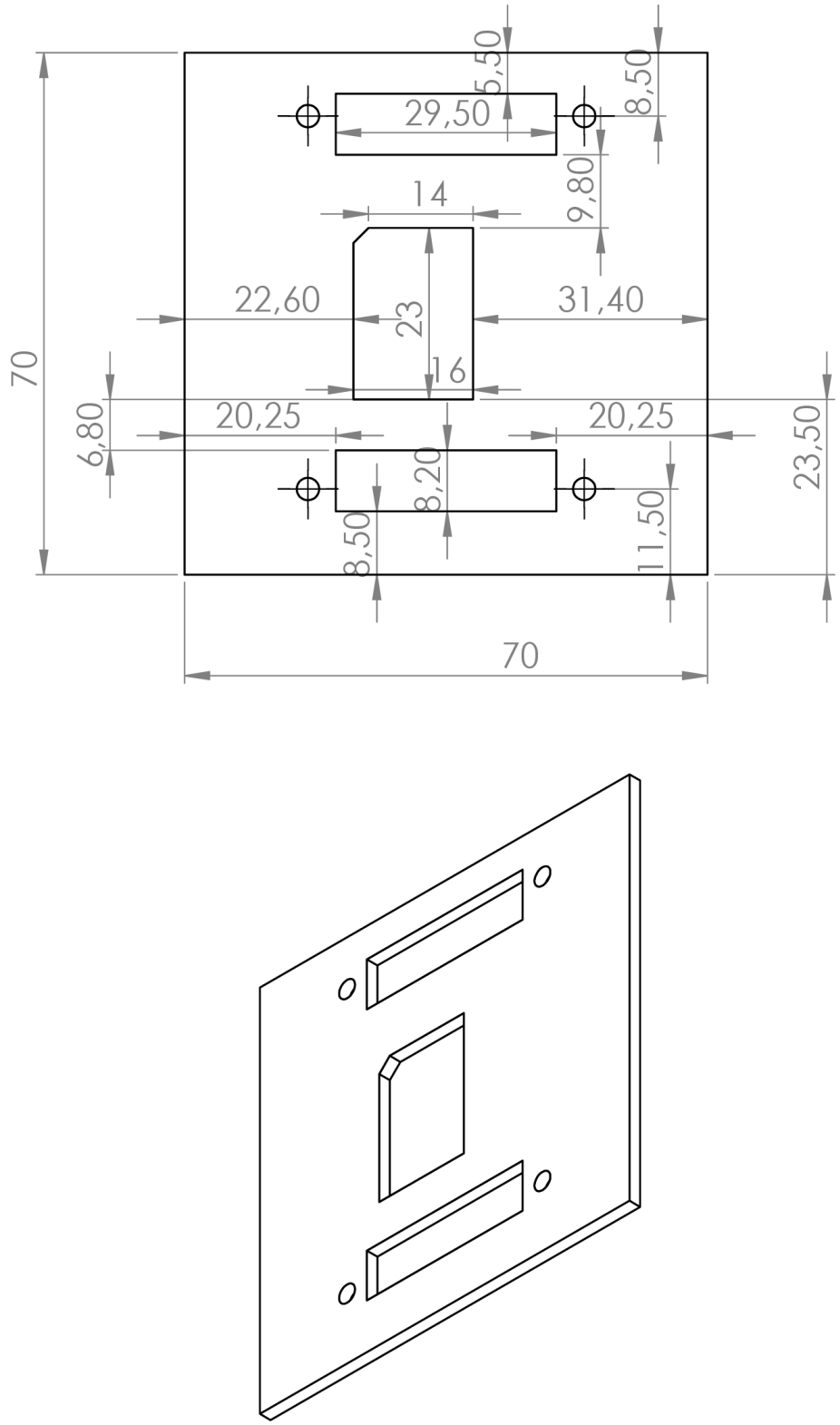
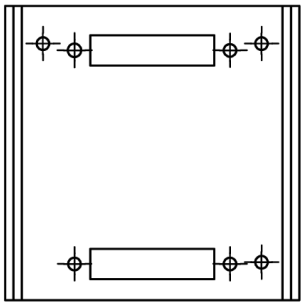
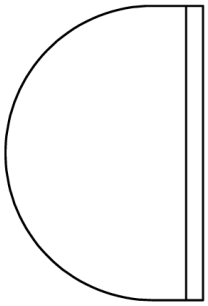
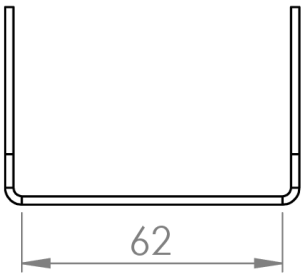
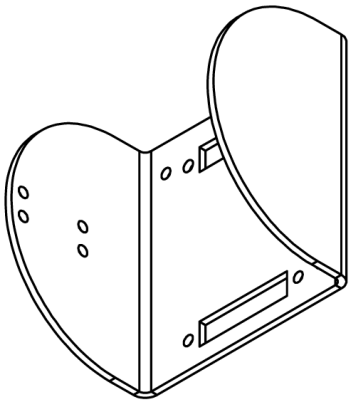


Figure IV.4: Cover Plan.









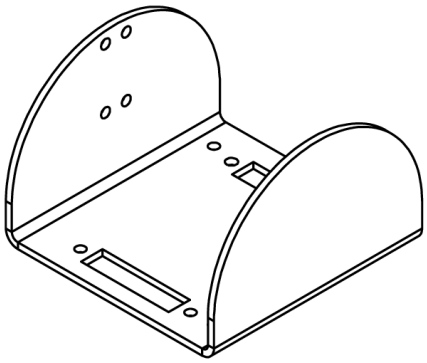


Figure IV.5: Base Blends.

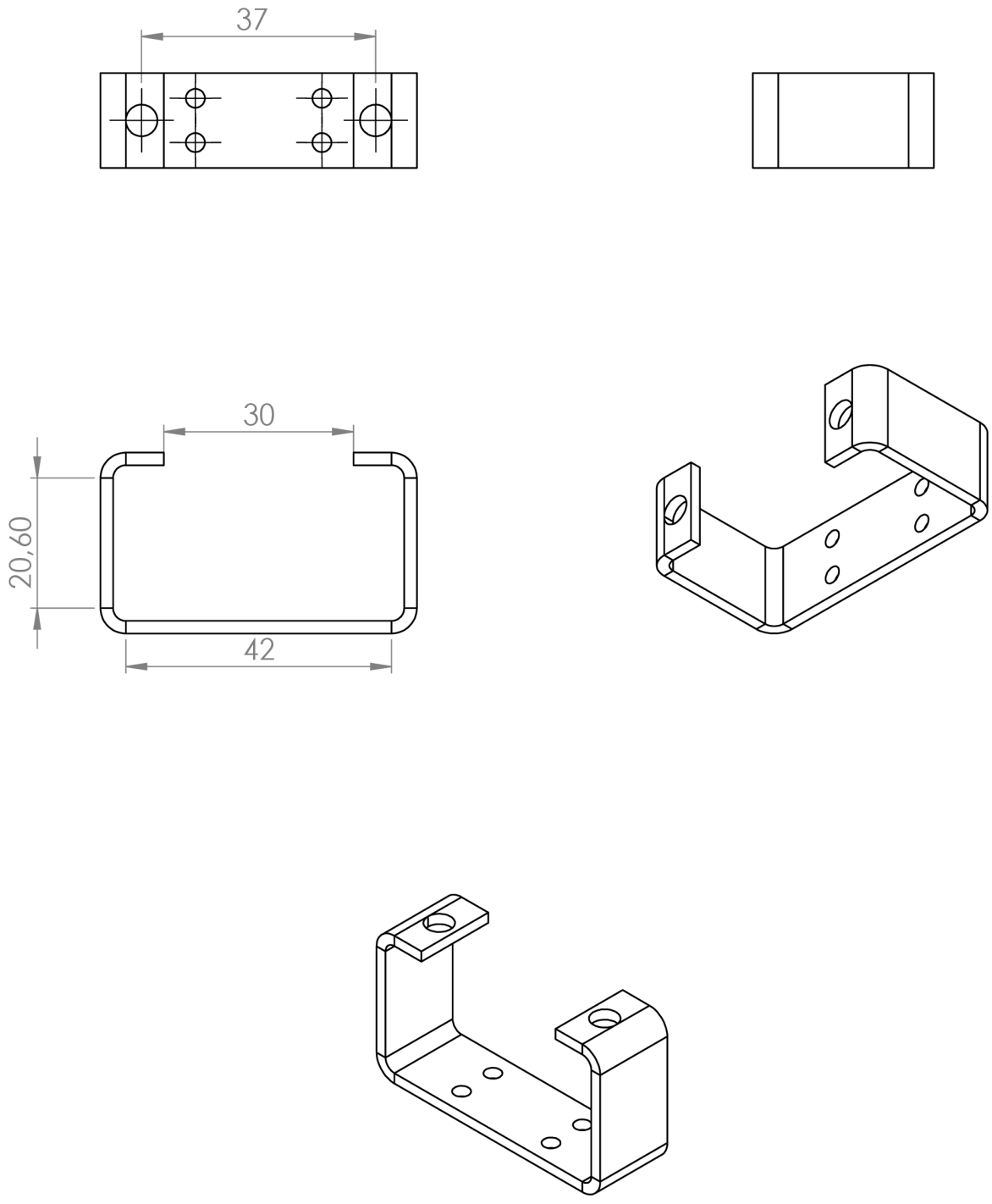


Figure IV.6: Motor Holder Blends.

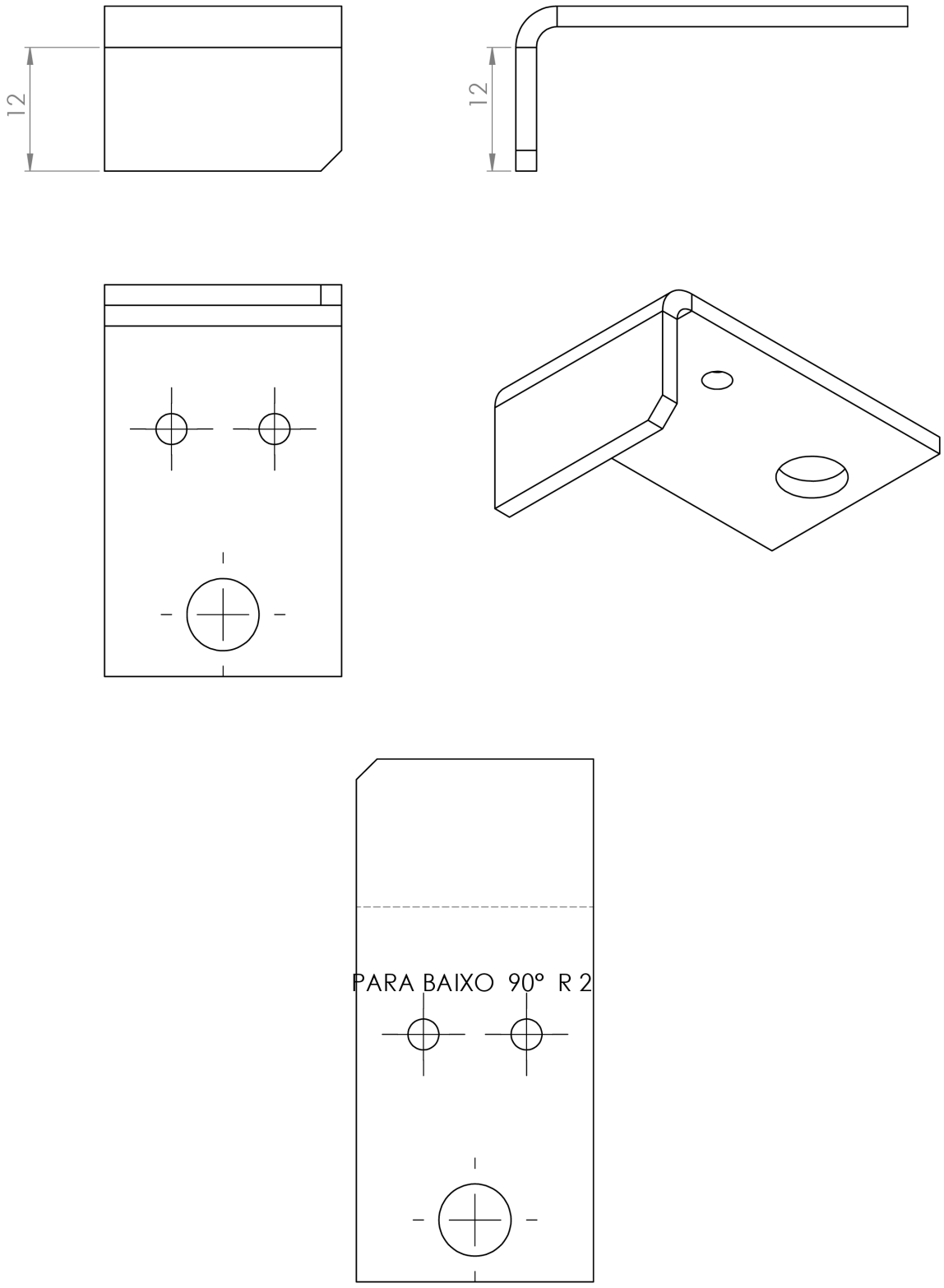


Figure IV.7: Rod Blends.

V. OVERVIEW OF THE ATTACHED DISK

V.1 ATMega8 Files

Codes in C language compiled by avr-gcc in Ubuntu 12.04.

atmega8-libraries.zip Basic libraries for the features of *ErekoBot* σ described in Appendix II.

V.2 Eagle Files

Files designed in CadSoft EAGLE PCB Design Software v6.5.0.

Figures from the board Figures for the electronic circuit of the module described in Appendix III.

eagle-schemes.zip Basic schematic for the electronic circuit of the module described in Appendix III.

V.3 Matlab Files

Routines and functions written in Matlab R2013a.

matlab-experiments.zip Basic routines and functions for the features of the module described in Section 5.

V.4 Media

ErekoBot σ figures and videos.

figures.zip Figures.

videos.zip Videos.

V.5 Report Files

Latex files compiled by pdfLatex.

dissertation.zip Latex files for compiling this report.

dissertation.pdf PDF file of this dissertation.

figs.zip Figures used in this report and in other articles.

V.6 SolidWorks Files

Files designed in CAD Software SolidWorks 2013.

solid-design.zip Basic design for the complete module described in Appendix IV.