Universidade Federal de Juiz de Fora

Faculdade de Engenharia

Engenharia Elétrica - Habilitação em Robótica e Automação Industrial

Henrique Magno Ferreira Longatti

Controle assistivo de um manipulador robótico articulado

Juiz de Fora

2016

Henrique Magno Ferreira Longatti

Controle assistivo de um manipulador robótico articulado

Trabalho apresentado à disciplina de Trabalho de Conclusão de Curso do curso de graduação de Engenharia Elétrica, habilitação em Robótica e Automação Industrial, da Universidade Federal de Juiz de Fora.

Orientador: Prof. Dr. Leonardo Rocha Olivi

Coorientador: M.Eng. Elias Ramos Vilas Boas

Juiz de Fora

2016

Ficha catalográfica elaborada através do programa de geração
automática da Biblioteca Universitária da UFJF,
com os dados fornecidos pelo(a) autor(a)

Longatti, Henrique Magno Ferreira.
Controle assistivo de um manipulador robótico articulado / Henrique Magno Ferreira Longatti. -- 2016.
58 f.

Orientador: Leonardo Rocha Olivi
Coorientador: Elias Ramos Vilas Boas
Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Juiz de Fora, Faculdade de Engenharia, 2016.

1. Robótica assistiva. 2. Manipulador robótico. 3. Controle compartilhado. 4. Otimização computacional. 5. Cinemática inversa. I. Olivi, Leonardo Rocha, orient. II. Vilas Boas, Elias Ramos, coorient. III. Título.

Henrique Magno Ferreira Longatti

Controle assistivo de um manipulador robótico articulado

Trabalho apresentado à disciplina de Trabalho de Conclusão de Curso do curso de graduação de Engenharia Elétrica, habilitação em Robótica e Automação Industrial, da Universidade Federal de Juiz de Fora.

ATA DE APRESENTAÇÃO DE TRABALHO DE CONCLUSÃO DE CURSO

ATA No. /2016, DA APRESENTAÇÃO PÚBLICA DE TRABALHO FINAL DE CURSO.

DATA DA DEFESA: 11/03/2016
CANDIDATO: HENRIQUE MAGNO FERREIRA LONGATTI
ORIENTADOR: LEONARDO ROCHA OLIVI
CO-ORIENTADOR: ELIAS RAMOS VILAS BOAS

BANCA EXAMINADORA:
LEONARDO ROCHA OLIVI / UFJF
ELIAS RAMOS VILAS BOAS / UFJF
ANA SOPHIA CAVALCANTI ALVES VILAS BOAS / UFJF
EXUPERRY BARROS COSTA / UFJF
LUCAS CORRÊA NETTO MACHADO / UFJF

TÍTULO DO TRABALHO: CONTROLE ASSISTIVO DE UM MANIPULADOR ROBÓTICO ARTICULADO

LOCAL: FAC. DE ENGENHARIA, ED. ITAMAR FRANCO, SALA 5205 HORA DO INÍCIO: 13h09

Em sessão pública, após exposição de cerca de 25 minutos, o candidato foi arguido oralmente pelos membros da banca, tendo o mesmo sido:

(X) APROVADO.
( ) APROVADO, COM A CONDIÇÃO DE SATISFAZER AS EXIGÊNCIAS CONSTANTES NA FOLHA DE MODIFICAÇÕES NO PRAZO DE _____ DIAS FIXADO PELA BANCA, E NÃO SUPERIOR A 30 (TRINTA) DIAS.
( ) REPROVADO.

Na forma regulamentar foi lavrada a presente ata que é abaixo assinada pelos membros da banca na ordem acima determinada e pelo candidato.

Juiz de Fora, 11 de MARÇO de 2016.

PRESIDENTE: Leonardo Rocha Olivi
Elias Ramos Vilas Boas
Ana Sophia Cavalcanti Alves Vilas Boas
Exuperry Barros Costa
Lucas Corrêa Netto Machado
CANDIDATO: Henrique Longatti

Vistos: Exuperry Barros Costa
Presidente da Comissão de TCC

Coordenador de Curso
Leonardo Rocha Olivi
Coordenador da Engenharia Elétrica
Robótica e Automação Industrial
Faculdade de Engenharia - UFJF

RESUMO

A área de robótica assistiva tem crescido de maneira acelerada, proporcionando diversas alternativas a pessoas que possuem, por algum motivo, dificuldades e imparidades de locomoção, de maneira a restaurar uma parte da mobilidade dessas pessoas, tornando sua vida e inclusão social mais facilitadas. Diversos trabalhos têm sido desenvolvidos na área de robótica móvel, com cadeiras de rodas robóticas, e manipuladores robóticos, com diversos graus de liberdade e diferentes geometrias, de modo a serem controlados por interfaces humano-máquinas assistivas, alimentadas por sinais biológicos dos usuários, como eletromiografia e eletroencefalografia. Este trabalho apresenta a construção de um manipulador robótico do tipo articulado, com três graus de liberdade, baseado em um robô industrial, a partir de Arduino e servomotores comerciais. Para o controle deste robô, propõe-se uma técnica de controle compartilhado baseado em campos potenciais de modo a reduzir a quantidade de esforço cognitivo gerado pelo usuário, reduzindo a quantidade de comandos dados para cumprir uma tarefa. Além disso, toda a parte do cálculo das cinemáticas direta e inversa por meio de Denavit-Hartenberg e Gradiente Descendente, respectivamente, são desenvolvidas em tempo computacional hábil. Os resultados mostram que a técnica é eficaz, capaz de resolver os problemas de restrições construtivas do robô e calcular parâmetros de juntas em tempo real, reduzindo a quantidade de esforço cognitivo de um usuário, auxiliando-o nas tarefas de controle compartilhado do manipulador com acurácia e segurança.

Palavras-chave: Robótica assistiva. Manipulador robótico. Controle compartilhado. Otimização computacional. Cinemática inversa. Cinemática direta.

ABSTRACT

The assistive robotics field has grown at an accelerated pace, providing plenty of options for people who have, for some reason, difficulties and impairments of movement, in order to restore a part of the mobility of these people, making their lives social inclusion easier. Several works have been developed in the mobile robotics field, such as robotic wheelchairs, and robotic manipulators area, with several degrees of freedom and different geometries, to develop new approaches of control by assistive human-machine interfaces, feed by biological signals of users, such as electromyography and electroencephalography. This paper presents the construction of an articulated robotic manipulator with three degrees of freedom, based on an industrial robot, built with Arduino and commercial servomotors. To control this robot, a shared control technique based on potential fields is proposed to reduce the cognitive effort that must be generated by the user, reducing the amount of commands to accomplish a task. In addition, the calculation of direct and inverse kinematics through Denavit-Hartenberg and descending gradient, respectively, are developed with sufficient computational time to be run in real time. The results show that the technique is effective, able to solve the robot building constraints and calculate joint parameters in real time, reducing the user's total cognitive effort, which is assisted by the shared control to handle the manipulator's tasks which accuracy and security.

Keywords: Assistive Robotics. Robotic Maniputator. Shared control. Computer optimization. Inverse kinematics. Forward kinematics.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 Introdução

A mobilidade é algo essencial na vida de todas as pessoas. A liberdade de se locomover e interagir com outros seres vivos e ambientes é uma essencialidade inerente na sociedade, de um modo individual e coletivo. Impedimentos físicos podem ser cometidos de diversas maneiras, impondo dificuldades às pessoas portadoras de deficiências. A justificativa deste trabalho encontra-se na possibilidade de devolver a essas pessoas portadoras de dificuldades motoras uma parcela de mobilidade por meio da robótica assistiva, tornando suas vidas mais independentes e com maior motivação.

Conforme aponta a pesquisa feita pelo IBGE [6], 6,8% da população feminina apresentando deficiência motora contra 4,5% da população masculina. As perguntas foram elaboradas para identificar as deficiências visual, auditiva e motora com seus respectivos níveis de seriedade. Uma análise mais geral obteve um índice de 26,6% de mulheres que têm pelo menos uma das deficiências citadas. Esse percentual corresponde a mais de 25,8 milhões de mulheres em todo o Brasil. Já o índice masculino é de 21% que corresponde a mais de 19,8 milhões de indivíduos. Uma parcela considerável da população nacional. Somente no Estado de Minas Gerais, 7,04% da população, cerca de 1,38 milhões de pessoas, possuem algum tipo de deficiência motora.

As deficiências motoras podem ter seu fator gerador de diversas maneiras. Doenças como esclerose múltipla e osteoartrite podem gerar deficiências motoras graves e até mesmo imobilizar a pessoa portadora por completo, bem como malformações congênitas. Além desses fatores patológicos existem os elementos causadores externos, como a perda de membros, traumas na coluna vertebral, acidentes vasculares, e assim por diante.

Existem alguns casos em que a deficiência pode ser tradada com sessões de fisioterapia, e até mesmo ser totalmente reparada. Mas em outros casos a pessoa fica confinada em uma cadeira de rodas pelo resto da vida. E isto leva a uma série de outros problemas, como depressão, dificuldade de socialização, a não aceitação no mercado de trabalho, a falta de infraestrutura em quase todos os setores da sociedade, impossibilitando assim a educação, lazer e esporte. Uma das funções da robótica, principalmente na área assistiva, é oferecer algum tipo de conforto a essas pessoas, proporcionando assim a possibilidade de realizar alguma de suas tarefas cotidianas, como por exemplo se locomover, pegar objetos, tomar decisões sem a ajuda de terceiros, tendo uma vida mais independente.

Com o avanço da tecnologia, as pessoas estão cada vez mais ligadas às máquinas. A comunicação cresce em rapidez e eficiência, proporcionando uma interação mais direta entre as pessoas por meio de seus dispositivos eletrônicos. Essa enorme interação está inundando as redes e nuvens eletrônicas com uma infinidade de dados que poderão ajudar as pessoas portadoras de deficiências. Por meio de análises de padrões para construção de sistemas baseados em sugestões, controles compartilhados e ambientes inteligentes,

é possível melhorar significativamente a vida das pessoas, especialmente aquelas com necessidades especiais [7, 8, 9].

Para que exista a comunicação entre pessoas e seus dispositivos eletrônicos, é necessária a existência de uma interface. As Interfaces Humano-Máquina (IHM, do inglês *Human-Machine Interfaces*) são peças de *software* responsáveis por realizar a comunicação entre o humano e a máquina. Para o caso específico deste trabalho, a operação das IHMs será por meio de sinais biológicos, como Eletroencefalograma (EEG) e Eletromiograma (EMG), em que estas são especificamente designadas para pessoas com deficiências, sendo chamadas de Interfaces Cérebro-Computador (BCI, do inglês *Brain-Computer Interface*), possibilitando a comunicação entre uma pessoa desprovida de mobilidade com a máquina. Com isso, o indivíduo é capaz de mover uma cadeira de rodas, movimentar um braço robótico, tomar decisões e entre outras coisas que tornariam sua vida um pouco mais independente.

A IHM assistiva que utiliza sinais cerebrais é feita por meio de EEG, e a que utiliza movimentos musculares é feita por sEMG (Eletromiografia de superfície, não invasiva), sendo estas as metodologias mais utilizadas, embora existam outras. Tanto o EEG quanto o sEMG utilizam os mesmos eletrodos de superfície não invasivos. Os sinais coletados por sEMG são referentes a sinais elétricos produzido pelas células nervosas no estímulo de um músculo em que o sensor está localizado, e é adequado para pessoas que possuem alguns movimentos corporais preservados, como os faciais. O EEG é usado quando a pessoa está totalmente impossibilitada de qualquer tipo de movimentação, como no caso da tetraplegia severa. Nesse tipo de situação, coletam-se sinais cerebrais padronizados, geralmente obtidos por meio de treinamento do usuário ou por sua resposta natural a estímulos externos, e utiliza-se classificação para a obtenção de comandos para os dispositivos de controle. Dentre alguns dos métodos mais populares de EEG-BCI estão o Potencial Evocado de 300 milissegundos, conhecido por P300, e o *Steady state visually evoked potential* (SSVEP) [10].

Neste trabalho, uma IHM assistiva, que utiliza EMG por meio do dispositivo Emotiv [1], irá se comunicar com um manipulador robótico de três graus de liberdade do tipo esférico, em que todas suas juntas são rotacionais. O braço robótico foi projetado e construído por meio de servo-motores controlados por meio do microcontrolador Arduino nano. A Figura 1 mostra o robô utilizado para este trabalho assistivo.

O equacionamento cinemático foi realizado por meio do método de Denavit-Hartenberg, um método clássico de equacionamento de cinemática direta de manipuladores. Para a obtenção dos parâmetros de juntas do robô, em resposta aos controles advindos do usuário por meio da IHM assistiva, utilizou-se o método de otimização quadrática do Gradiente Descendente. O planejamento dos caminhos entre os pontos escolhidos pelo usuário foi feito por meio do método de Campos Potenciais, evitando desta maneira,

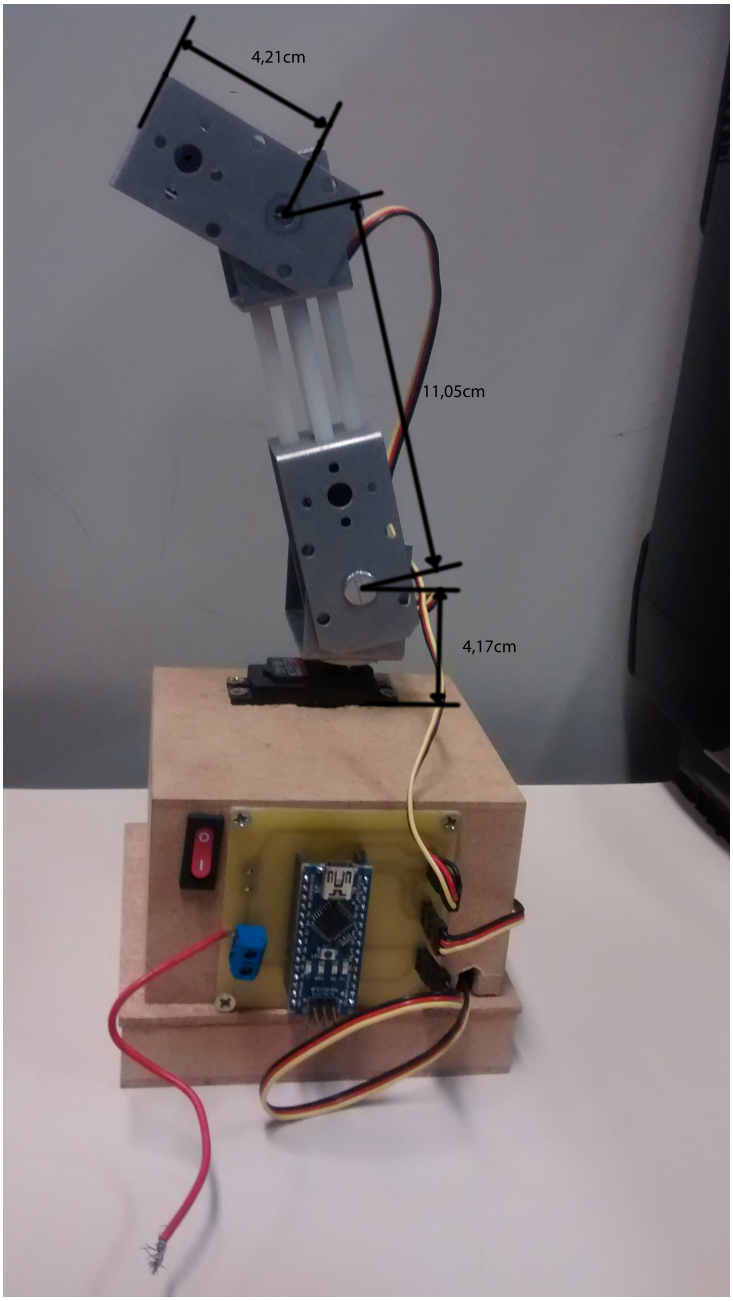

Figura 1 – Robô esférico utilizado neste trabalho.

possíveis colisões, em que os dados do ambiente são feitos por meio de sensores *rangefinders*, como o SONAR. Com essas ferramentas o usuário é capaz de realizar a teleoperação de um manipulador robótico utilizando a IHM com sinais de sEMG.

A descrição do trabalho, além da introdução descrita neste capítulo, segue a seguinte ordem: No Capítulo 2 é feita uma revisão bibliográfica para a monografia, mostrando diversos trabalhos correlatos presentes na literatura. No Capítulo 3 apresenta-se a fundamentação técnica e descrição dos diversos conceitos utilizados no trabalho. No Capítulo 4, os resultados de simulação e experimentos com robô real são mostrados, bem como sua análise. No Capítulo 5 tem-se a conclusão e descrição de trabalhos futuros.

2 Revisão Bibliográfica

Este capítulo apresenta a revisão bibliográfica da literatura concernente a trabalhos correlatos ao proposto nesta monografia. Além da revisão, mostram-se também conceitos relativos à robótica assistiva e também a conceituação técnica do trabalho desenvolvido. Este capítulo envolverá a análise das propostas de interface humano-máquina e também trabalhos em robótica assistiva na área de controle compartilhado.

2.1 Robótica Assistiva

A robótica assistiva surgiu para atender uma grande quantidade de pessoas com algum tipo de deficiência motora. No caso, esse grupo de pessoas tem acesso a essa nova tecnologia para ajudar em suas tarefas motoras diárias. Ela teve seu crescimento a partir da década de 1990, devido ao desenvolvimento computacional que resultou numa maior velocidade na comunicação, processamento, dentre outras vantagens [10, 11].

A área assistiva floresceu por meio de contribuições importantes, como de N. Papanikolopoulos et al. (1995) [12] e de K. Nagai et al. (1998) [13]. O primeiro artigo aborda o problema da combinação do controle com a visão computacional para o desenvolvimento de atividades baseadas e assistidas por meio de imagens. Já o segundo é sobre um manipulador robótico de 8 graus de liberdade (DOFs, do inglês *Degrees of Freedom*), que corresponde ao numero de variáveis independentes de posição que precisa ser especificadas para se definir a localização de todas as partes articuladas do Manipulador. Ele foi construído para ajudar a movimentação do antebraço de um humano.

No início dos anos 2000 o desenvolvimento da tecnologia assistiva caminhou junto com o da computação. Por meio do ganho de memórias expandidas, proporcionando uma maior capacidade de armazenamento, e também da evolução dos processadores, o avanço da computação possibilitou o desenvolvimento de novas técnicas semiautônomas e manuais, uma vez que em décadas anteriores, a maior parte das técnicas eram autônomas [10, 11].

A evolução da computação também beneficiou os aparelhos usados nas ciências médicas, tornando possível a melhor capitação e processamento de sinais sEMG e EEG. Isto possibilitou o desenvolvimento dos trabalhos como o do M. Palankar et al. (2009) [14], controlando um braço robótico com 9 graus de liberdade, I. Pathirage et al. (2013) [15], usando visão computacional para reconhecimento de objetos e tomada de decisão sobre a utilização da garra de um manipulador robótico, e K. George et al. (2014) [16], desenvolvendo uma interface humano-máquina assistiva para a interação de uma pessoa com um manipulador robótico.

Diversos desses trabalhos empregam manipuladores fixados em bases robóticas móveis, usualmente no formato de cadeiras-de-rodas robotizadas, permitindo que um

usuário com impedimentos motores possa se locomover pelos ambientes de acordo com sua vontade, recuperando assim sua mobilidade. Além disso, esse usuário utiliza o manipulador robótico para interagir com os objetos deste ambiente. Todos esses processos são possíveis por meio da aquisição de comandos do usuário via interfaces humano-máquinas assistivas, muitas vezes constituídas por interfaces cérebro-computador ou por sEMG [10, 11, 14, 15, 16].

2.2 Interface humano-máquina assistiva

O objetivo principal no desenvolvimento de uma interface humano-máquina assistiva é realizar uma comunicação entre o usuário e o sistema robótico, o qual proverá auxílio ao usuário na realização de uma determinada tarefa. Essas interfaces devem ser capazes de obter comandos advindos do usuário por meio de seus sinais biológicos e processos de aprendizado de máquina.

Para adquirir essa capacidade de extração, diversas técnicas de aprendizado de máquina e classificação de sinais são desenvolvidas por causa da individualidade de cada um dos usuários, que podem ser susceptíveis a certos métodos mais que outros, tornando-os eficazes em determinados contextos. Na literatura encontram-se diversas técnicas de realizar comunicação humano-maquina assistiva. As mais usadas são as baseadas em Eletroencefalografia (EEG) e Eletromiografia de superfície (sEMG), as quais são não-invasivas [10].

Um conceito importante relacionado a sinais biológicos é se a técnica de aquisição é feita de maneira invasiva ou não-invasiva. A técnica invasiva recebe os sinais diretamente do sistema nervoso e necessita de meios cirúrgicos para coletá-los, uma vez que eletrodos são implantados dentro do corpo do usuário, diretamente no local de coleta. Já a técnica não-invasiva recebe os dados por meio de sensores instalados sobre a epiderme do usuário, sem que exista qualquer tipo de procedimento invasivo. Essa capitação é feita por eletrodos que podem ser molhados (*wet electrodes*), ou seja, que dependem de soluções condutoras de eletricidade, ou eletrodos secos (*dry electrodes*), independentes de quaisquer soluções condutoras. As técnicas usadas nesse trabalho são todas não-invasivas.

Conforme mencionado, a comunicação entre o usuário com algum tipo de deficiente motora se faz por meio de sinais biológicos. O sistema nervoso é capaz de produzir sinais elétricos através de complexas atividades eletroquímicas celulares. Esses sinais são responsáveis por controlar movimentos voluntários e involuntários de órgãos e músculos [17]. A maioria das aplicações utiliza EEG e sEMG, que são sinais elétricos, sendo realizados por meio de eletrodos, que são sensores relativamente pouco custosos e de fácil aquisição mercadológica.

Métodos de obtenção, além do EEG e do sEMG, são a variação do fluxo sanguíneo,

obtida por meio de espectroscopia de reflectância no infravermelho-próximo (NIRS, do inglês *Near Infrared Reflectance Spectroscopy*) e a captação de imagens internas do corpo e cérebro, por meio de ressonância magnética funcional (fMRI, do inglês *functional Magnetic Resonance Imaging*), dentre outros. Quando comparados com o EEG e o sEMG, estes outros métodos são muito mais custosos [10, 17].

As interfaces cérebro-computador (BCI) são desenvolvidas, primordialmente, por meio de sinais de eletroencefalografia (EEG). Ela é uma interface humano-máquina que tem a função de coletar e classificar alguns padrões cerebrais. As classificações dessas atividades cerebrais são de grande complexidade, pois existe uma infinidade de atividades e reações que o cérebro gera quando ele é estimulado. São indicadas para os usuários que estão completamente paralisados e, consequentemente, não possuem grupos musculares que possam auxiliar o processo de classificação.

Conforme a tecnologia em BCI avança, novos métodos de classificação de padrões celebrais são criados. Um dos métodos mais utilizados na literatura é o método do Potencial Evocado de 300 milissegundos, ou, P300. O sinal P300 é evocado por meio de uma reação ao um estímulo cerebral intencional do usuário, como uma contagem mental, e é colhido nos lobos occipital e parietal cerca de 300 milissegundos após o estímulo.

Existe também o método de Potenciais Visuais Evocados em Regime Estacionário (SSVEP, do inglês *Steady-State Visually Evoked Potentials*), no qual são gerados estímulos visuais na forma de uma matriz de opções, em que cada posição desta matriz corresponde a uma frequência específica. Quando o usuário foca sua atenção em uma das opções, o cérebro responde em uma frequência igual ou múltipla daquela sendo vista, a qual pode ser obtida no córtex visual. Esses estímulos podem ser luzes piscando em frequências de 3,5 a 100 Hz.

Diversas outras técnicas podem ser utilizadas em BCI, como a imaginação de movimento, em que o usuário faz esforço mental imaginando a movimentação de braços, pernas, língua, dentre outros, e o cérebro gera sinais específicos para as ações imaginadas. Aplicações de BCI têm tempos variados, dependendo da técnica utilizada, indo desde milissegundos até vários segundos, com índices de acerto satisfatórios dependendo da aplicação [10, 11, 17].

A técnica de sEMG coleta informações elétricas na movimentação de músculos corporais, e são indicadas para aqueles usuários que possuem algum grupo muscular funcional, como por exemplo a musculatura da face. O método se baseia em coletar os dados elétricos que percorrem os nervos e músculos quando um movimento é feito. Cada movimento muscular possui características específicas, denominadas por assinaturas ou artefatos, que possibilitam, por meio de algoritmos de classificação, efetuar a distinção entre diversos movimentos, como por exemplo, piscadas, sorrisos, mordidas, movimentos de sobrancelhas, e assim por diante.

Com diversos eletrodos de sEMG abrangendo uma área de interesse, o grupo muscular acionado pelo usuário irá emitir sinais elétricos captados pelos eletrodos localizados na vizinhança. Assim, por meio de técnicas de processamento de sinais é possível classificar o movimento feito, com filtragens, reconhecimentos de frequências, eliminação de ruídos, treinamento de classificadores como *Common Average Reference* (CAR) ou Redes Neurais Artificiais (RNA), dentre outros. Classificações de sEMG são rápidas, da ordem de milissegundos, sendo indicadas para atividades em tempo real, além disso, possuem alto índice de acertos para a grande maioria dos usuários [7, 10, 11, 17]. Tanto EEG como sEMG usam como base de entrada de dados os mesmos tipos de eletrodos.

2.2.1 Emotiv

O Emotiv EPOC/EPOC+ é um dispositivo comercial desenvolvido pela empresa Emotiv Systems [1] que se faz uma interface cérebro-computador baseada em EEG por meio de um capacete utilizando hastes fixas para os eletrodos, oferecendo tanto classificações por padrões cerebrais quanto por sEMG. Este produto possui 14 canais de EEG e 2 de referência, um sistema wireless de comunicação e transferência de dados, com baixo peso e baterias recarregáveis, sendo utilizado em diversas pesquisas no contexto assistivo, de entretenimento e neuroterapia [1]. O aparelho é mostrado na Figura 2.

Figura 2 – Epoc/Epoc+ [1].

O Emotiv é capaz de detectar tanto sinais EEG como sinais sEMG. O *Emotiv control Painel Software*, disponibilizado no site do produto [1], é usado para interpretar os sinais sEMG e classificá-lo por meio de um caracter do teclado, a ser configurado pelo usuário, e a interface que permite esta configuração é a *Emotiv Control Painel*, conforme mostra a Figura 3. Além da classificação por sEMG, o Emotiv possui outros módulos, o *Expressiv* e o *Cognitiv*. O *Expressiv* emula as expressões do usuário e o *Cognitiv* identifica os padrões de sincronização e dessincronização de concentração, conhecido como ERD, do inglês, *Event Related Desynchronisation*. Além disso, oferece também treinamento e

classificação para a utilização de comandos de imaginação de movimentos via BCI, como empurrar objetos, rotacionar, dentre outros.

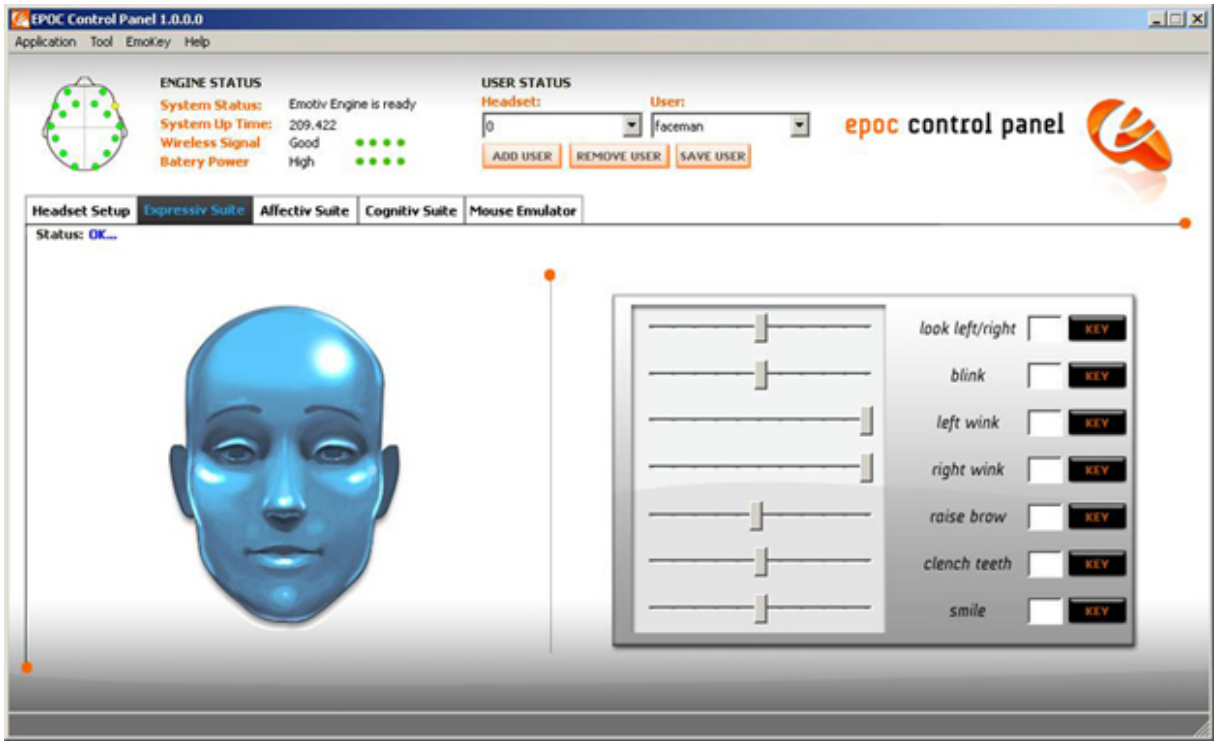

Figura 3 – *Emotiv control panel* [1].

No contexto de sEMG, esse software é capaz de interpretar vários movimentos faciais, conforme pode-se notar na Figura 3, como por exemplo: levantar as sobrancelhas, piscar o olho direito, piscar o olho esquerdo, mordida, olhar para direita e olhar para esquerda. O tempo de resposta é relativamento rápido, na ordem de milissegundos. Sua taxa de sucesso na classificação, ou seja, sua probabilidade de acerto, varia entre os usuários, pois diversos fatores influenciam o funcionamento do aparelho, desde sua colocação, umidificação dos eletrodos, bem como as características intrínsecas e individuais de cada usuário. Um sistema tradicional EGG custa por volta de dezenas de milhares de dólares. O Emotiv tem seu preço acessível, cerca de algumas centenas de dólares, o que torna possível a comercialização desse dispositivo tanto para pesquisa quanto para um usuário comum. Por essas diversas características positivas este aparelho foi utilizado neste trabalho para fazer o interfaceamento humano-máquina entre o usuário e o robô. Ao usuário, usando o capacete (*headset*) Emotiv, cabe a geração de sinais de controle para o braço robótico mostrado na Figura 1, página 13.

2.3 Fundamentação técnica

O trabalho desenvolvido por M. Palankar et al. (2009) [14] se trata de uma cadeira de rodas equipada com um manipulador de sete graus de liberdade (DoFs, do inglês,

Degrees of Freedom). Ao todo o robô possui nove graus de liberdade. O seu controle vai além dos métodos convencionais, pois ele combina o controle da cadeira de rodas com o do braço robótico. Os autores realizam a junção dos Jacobianos que representam os sistemas de controle individualmente da cadeira e do braço robótico, resultando em um sistema de controle conjunto. Assim, o controlador projetado oferece robustez e tarefas otimizadas para combinar a sua mobilidade e manipulação ao mesmo tempo.

O sistema de controle foi projetado para ser teleoperado ou autônomo, em que a interface com o usuário é feita por meio de BCI com P300 para controlar o sistema WMRA (*Wheelchair Mounted Robotic Arm*). Dessa maneira, oferece-se ao usuário uma matriz visual que contém símbolos ou uma matriz alfabética, em que cada célula representa uma ação ou comando para o robô. As opções dessa matriz irão piscar aleatoriamente uma mesma quantidade de vezes cada. Por meio do BCI P300, o usuário portador de necessidades especiais é capaz de selecionar um dos comandos desejados da matriz apenas concentrando a sua atenção na sua opção de escolha e contando mentalmente todas as vezes que esta piscar.

Os autores de J. Webb et al. (2012) [18] tratam de um dispositivo robótico para auxiliar na reabilitação dos pacientes que sofreram algum tipo de acidente vascular cerebral. Consiste na construção de um sistema robótico portátil tipo exoesqueleto controlado por interface BCI. Os autores utilizaram um dispositivo comercialmente disponível, o Emotiv EPOC, para realiza a interface humano-máquina. Além da construção foi feito um estudo para avaliar o desempenho do sistema com quatro voluntários adultos saudáveis. O estudo pretende avaliar o desempenho da IHM no comando de um exoesqueleto que irá auxiliar pacientes em reabilitação. Assim podendo tornar o tratamento menos exaustivo tanto para o paciente quanto para a equipe de fisioterapia. O objetivo do trabalha é controlar exoesqueleto via BCI capaz de atuar como um cotovelo de uma pessoa saudável por meio da imaginação de movimentos de tal membro. O sistema usa apenas um grau de liberdade.

A pesquisa de [18] contou com quatro voluntários saudáveis, todos homens com idades entre 25 e 35 anos. Primeiramente os voluntários foram submetidos ao treinamento para o uso do Emotiv. Depois, utilizando o Emotiv e o sistema robótico, cada voluntário necessitou realizar um série de flexões e relaxamentos da junta do cotovelo. Conforme relatam os autores, os resultados tiveram uma média de acurácia aceitável viabilizando o projeto funcional como uma possibilidade de reabilitação pós acidente vascular cerebral.

O trabalho desenvolvido pelos autores de I. Pathirage et al. (2013) [15] demonstra uma nova abordagem BCI não-invasiva baseada em visão computacional para agarrar objetos usando um braço robótico através de um uma grade de estímulos visuais adaptativos para a seleção dos objetos dentro de uma imagem. O objetivo era reduzir a carga cognitiva do usuário assim com o tempo necessário para executar uma tarefa. A imagem é gerada através de uma câmera montada no sistema WMRA. Uma grade dinamicamente

dimensionada de estímulos BCI não uniforme é criada. Tal grade, conforme estabelecem os autores, melhora a eficiência da seleção dos objetos. O BCI P300 foi usado para selecionar as opções disponíveis ao usuário. Conforme apontam os autores, o resultado implicou em uma média de acurácia na seleção de 85,56% dentre todos os objetos disponíveis, com uma quantidade média de cinco comandos exercidos pelo usuário para agarrar o objeto.

O trabalho de N. Yamawaki et al. (2013) [19] trata do caso especial de pacientes com dificuldades motoras devido à esclerose lateral amiotrófica ou por lesões ocorridas na coluna vertebral, tendo como enfoque o movimento das mãos, essencial para executar tarefas rotineiras. Tendo disponível um manipulador robótico, seu controle foi feito por meio de BCI. O manipulador é equipado com um sensor ultrassônico (SONAR) e uma câmera. Por meio desses dispositivos, é possível evitar colisões entre o manipulador e o paciente, possibilitando um controle parcialmente automático, ou seja, um controle compartilhado entre o usuário e o sistema de controle. Em [19], o computador que controla o manipulador robótico recebe três tipos de dados em suas entradas: o sinal de saída do BCI, a imagem do objeto e a distância até o usuário. O BCI é usado para selecionar a ação a ser executada pelo manipulador, a imagem da câmera e a distância do SONAR são usadas para distinguir os objetos e para mover automaticamente o braço do robô para o mais perto possível do objeto alvo.

O artigo feito por K. Georger et al. (2014) [16] tem por objetivo viabilizar a tecnologia de BCI no controle de exoesqueletos ou em membros artificiais para pacientes em recuperação de alguma lesão muscular. O Emotiv foi usado para reconhecer os padrões de movimentos musculares e, então, convertê-los em um sinal digital eletrônico, para o qual uma interface em MATLAB® [20] se comunica com o processador Arduino® Uno [21] que controla os motores do exoesqueleto. Segundo os autores, o resultado experimental obteve um tempo médio de resposta a estímulos 9,18% menor usando o sistema controlado com BCI quando comparado com um sistema controlado sem BCI, ou seja, com um *joystick* normal, viabilizando o projeto com BCI para o controle de braços robóticos ou exoesqueletos.

Temos, portanto, que a preocupação na utilização de manipuladores robóticos por meio de interfaces humano-máquina assistivas é crescente, com diversos trabalhos na literatura desenvolvendo várias alternativas para tratar o problema e, consequentemente, melhorar a vida das pessoas. Tendo em vista a preocupação da área, este trabalho se propõe a apresentar uma alternativa de controle compartilhado assistivo de um manipulador robótico para pessoas com deficiência motora.

2.4 Contribuição deste trabalho

Esta monografia trata da construção e do controle compartilhado via BCI de um manipulador esférico de três graus de liberdade, em que todas as suas juntas são rotacionais,

conforme descrito no Capítulo 1 e na Figura 1, página 13. O trabalho tem a intenção de ajudar as pessoas que tem algum tipo deficiência motora nos braços a realizar algumas tarefas cotidianas, recuperando sua mobilidade. A princípio o usuário é equipado com o aparelho Emotiv EPOC, o qual disponibilizará alguns movimentos faciais para controle do robô, como por exemplo, morder, levantar as sobrancelhas, olhar para a direita e olhar para esquerda. Estes movimentos estarão associados a comandos diretos do robô, de forma que o usuário consiga teleoperá-lo para uma posição desejada. Juntamente com o usuário, o controle será compartilhado com um algoritmo baseado em Campos Potenciais [22], de forma a proteger o robô, e por extensão o usuário, durante os movimentos realizados.

Na literatura pode-se encontrar uma grande quantidade de soluções em se tratando de robótica assistiva no controle de manipuladores robóticos. Alguns autores tratam dos casos em que os robôs são equipados com sensores exteroceptivos, como câmeras e *rangefinders*, dispositivos que medem distância. Com esses sensores o manipulador é capaz de identificar objetos e planejar sua linha de ação com base neste conhecimento, levando sempre em consideração a segurança do usuário e proporcionando um controle menos exaustivo para o mesmo, reduzindo sua carga de esforço cognitivo.

O robô construído para este trabalho não possui sensores exteroceptivos, e, portanto, a percepção de obstáculos para a validação da estratégia de controle compartilhado se dará por meio de simulação. O robô real será utilizado para validar as estratégias proprioceptivas, como o cálculo de cinemática direta e inversa e controle de juntas para o posicionamento correto do robô.

A escolha de comandos do robô se dará por meio de um aparelho Emotiv EPOC utilizando sEMG. Dessa maneira, a interface humano-máquina desenvolvida prezou pela redução da carga cognitiva que o usuário terá que imprimir, com a redução da quantidade de movimentos permitidos ao robô. Os movimentos faciais do usuário se comportam como um *joystick* para posicionar a ferramenta do manipulador (*end-effector*). Para trabalhos futuros, tem-se a implementação de sensores exteroceptivos reais no robô, como câmeras e *rangefinders*, e com isso realizar um controle automático ou semiautomático para a posição do *end effector* com o robô real da mesma maneira que o mostrado nas simulações.

3 Fundamentação técnica

Manipuladores robóticos, em geral, são máquinas eletro-mecânicas inspiradas na anatomia humana, contendo braço, antebraço, cotovelo, e assim por diante. Os elos, ou *links*, são as partes fixas inspiradas no braço e antebraço. As juntas são as partes móveis, que contém elementos que efetivam movimentos, como motores, e são inspiradas no punho, cotovelo e ombro.

Existem diversos tipos de juntas disponíveis para a construção de robôs manipuladores. No entanto, todas elas podem ser simplificadas pelo arranjo de dois tipos: rotacionais e prismáticas. Essas juntas possuem um grau de liberdade (DoF, do inglês *Degree of Freedom*) cada uma, ou seja, têm a liberdade de agir em apenas uma dimensão disponível, a do eixo de seu único motor. Dessa maneira, as juntas rotacionais ou prismáticas necessitam de apenas uma variável para determinar completamente seu estado, posição angular ou linear, respectivamente.

Esses conceitos são importantes para o estudo das cadeias cinemáticas que compõem os robôs manipuladores, tornando-os mecanismos com movimentos complexos construídos a partir de elementos capazes de executar movimentos simples, as juntas. Existem diversos manipuladores disponíveis no mercado, contendo desde poucos graus de liberdade até manipuladores de 6 ou mais graus de liberdade. A seguir, o capítulo trata dos diversos conhecimentos envolvidos no desenvolvimento deste trabalho.

3.1 Cinemática

A cinemática é a ciência que estuda o movimento de objetos sem a preocupação de descrever suas causas. A cinemática preocupa-se em descrever movimentos com relação a referenciais (*frames*) bem definidos, trabalhando principalmente com objetos pontuais, como pontos, retas, parábolas, e assim por diante. Em diversos livros-texto oferecidos na literatura, a cinemática é chamada de “a geometria do movimento”. Um robô manipulador é constituído por inúmeras peças conjuntas, como juntas, elos (ou *links*), garras e sensores. No entanto, ainda que esses elementos sejam constituídos de massas, a cinemática os simplifica de modo a representá-los por pontos e retas.

Esses pontos e retas que representam o robô se deslocarão no espaço de maneira organizada pela geometria construtiva do robô, a qual depende totalmente do projeto construtivo. Uma vez dispondo dessas informações construtivas, é possível estabelecer sistemas de referência apropriados para que os pontos e retas que constituem a representação do robô possam realizar as translações e rotações que possibilitam a representação da movimentação real do robô. De maneira generalista, *frames* são, de fato, eixos de coordenadas espaciais usados para medir grandezas físicas como distâncias, velocidades, acelerações, campos eletromagnéticos, e assim por diante.

Conforme mencionado, um robô manipulador é composto por elos (*links*) e juntas, que intercalados entre si formam o robô de acordo com o projeto construtivo. Dependendo da maneira de interconexão entre os elos e juntas, o robô poderá realizar movimentos diversificados ou não, sendo esta uma restrição construtiva. Em um extremo do robô encontra-se a base, a parte do robô presa a uma superfície de sustentação. No outro extremo encontra-se o punho do robô, ao qual podem ser conectados os mais diversos tipos de ferramentas (*end-effectors*), como mãos robóticas, garras, ferramentas de solda, pintura, furadeiras, parafusadeiras e assim por diante. As ferramentas dependem da função do robô em seu local de aplicação.

Todas as partes do robô possuem seus próprios sistemas de coordenadas (*frames*), e, portanto, serão estudadas por meio das descrições e manipulações desses sistemas, com translações e rotações. Esses diversos *frames* são evidenciados na Figura 4. Nas translações, seus eixo coordenados se deslocam de maneira linear pelo ambiente ao longo de um ou mais eixos coordenados, possibilitando a mudança de posição sem a alteração da orientação. Nas rotações, ocorre o deslocamento angular com relação aos eixos coordenados, as quais são comumente denominadas de *roll*, *pitch* e *yaw* os ângulos de rotação nos eixos x, y e z, respectivamente, e são mostradas na Figura 5 [23]. As próximas seções descreverão esses movimentos e seus equacionamentos.

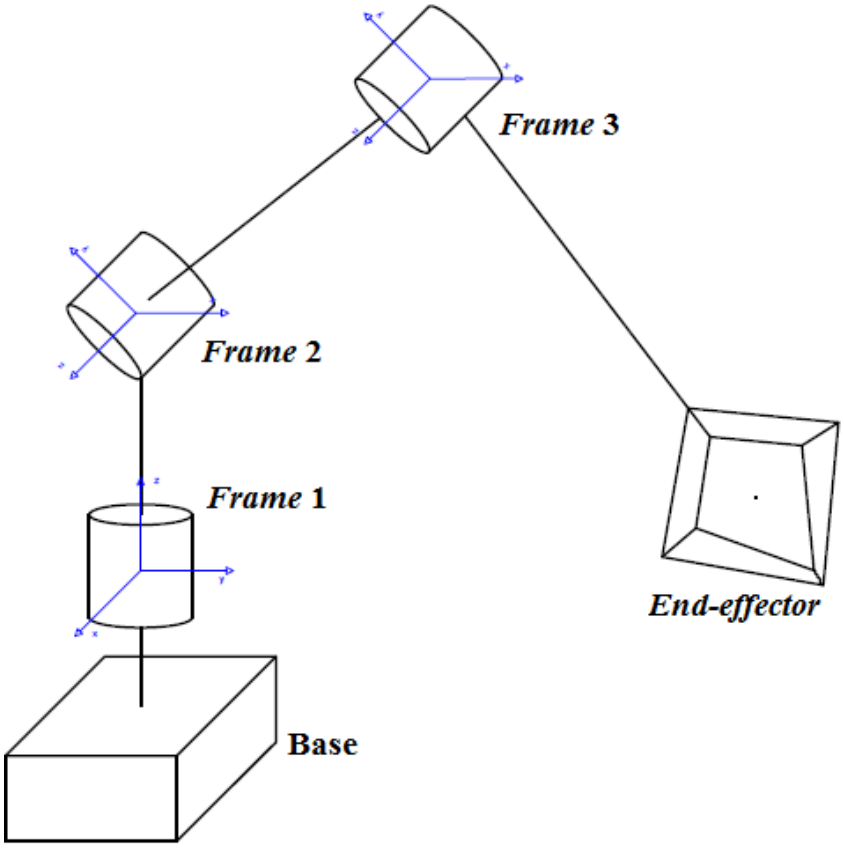

Figura 4 – Sistemas de coordenadas (*frames*).

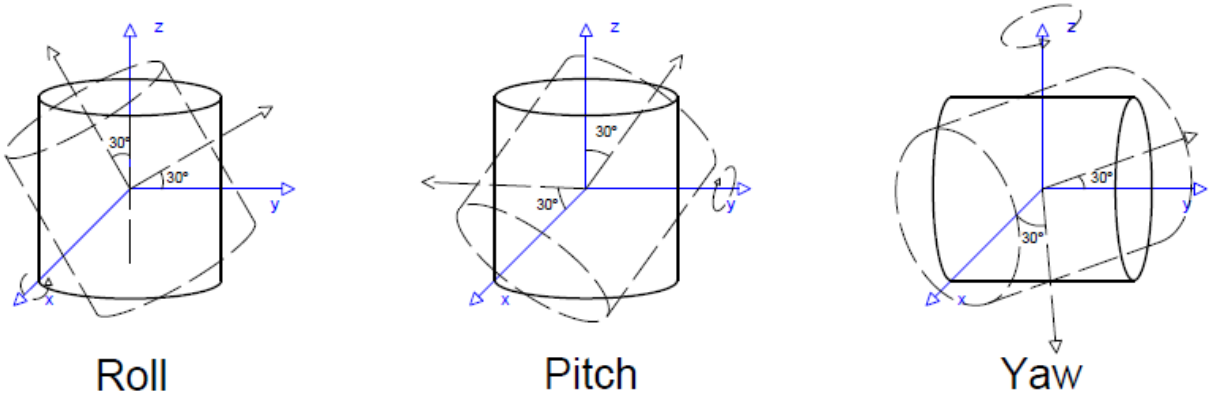

Figura 5 – Rotações: *roll*, *pitch* e *yaw*.

3.1.1 Translação

Em um sistema de coordenadas, quando deseja-se que um corpo rígido se desloque no espaço de uma quantidade Δs pré-definida, esta quantidade é decomposta e acrescentada, de maneira linear, em cada uma das variáveis que descrevem seu sistema de coordenadas (*frame*). As equações da translação bidimensional são mostradas a seguir, e um exemplo é mostrado na Figura 6.

$$\begin{aligned} x_1 &= x_0 + \Delta x \\ y_1 &= y_0 + \Delta y \end{aligned} \tag{3.1}$$

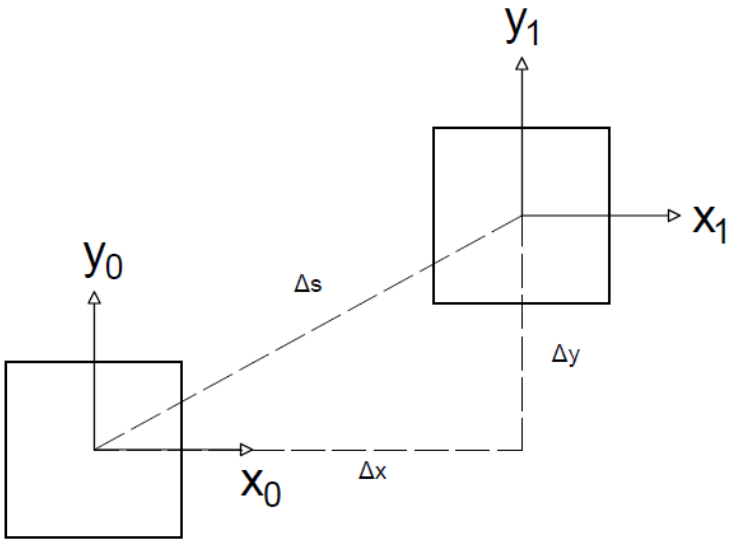

Figura 6 – Translação bidimensional de um corpo rígido.

É conveniente, especialmente em robótica, trabalhar por meio de matrizes, uma vez que as operações matriciais facilitam a manipulação de variáveis em sistemas multidimensionais. Para o caso de robôs manipuladores tem-se a seguinte releitura das equações mostradas em 3.1 em formato matricial, apresentada em 3.2.

$$x_1 = x_0 + \Delta x = 1.x_0 + 0.y_0 + \Delta x$$
$$y_1 = y_0 + \Delta y = 0.x_0 + 1.y_0 + \Delta y \tag{3.2}$$

Nota-se que as quantidades Δx e Δy, que adiciona a translação ao sistema de coordenadas, precisam de uma dimensão extra para serem encapsulados em uma operação matricial. Dessa maneira, acrescenta-se uma dimensão extra no vetor das coordenadas iniciais $\{0\}$ de forma a abarcar as quantidades de deslocamento. Essa nova dimensão mostra que deslocamentos bidimensionais acontecem num espaço matricial tridimensional, mostrado nas equações subsequentes, tanto para o *frame* $\{0\}$ quanto para o $\{1\}$ [22, 23].

$$\mathbf{X_0} = \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.3}$$

$$\mathbf{X_1} = \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{3.4}$$

Organizando as equações em formato matricial tem-se:

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.5}$$

que pode ser reescrita como:

$$\mathbf{X_1} = \mathbf{T}\ \mathbf{X_0} \tag{3.6}$$

A matriz $\mathbf{T}$ é chamada de matriz de translação. Ela é responsável por deslocar linearmente um corpo rígido em um ambiente. Porém, os manipuladores não trabalham somente em um plano bidimensional. A medida que agregamos mais graus de liberdade na construção de um robô, sua complexidade aumenta. Um robô com muitos graus de liberdade pode trabalhar em um espaço tridimensional, desde que seus elos e juntas não estejam organizados de forma planar. Em um espaço tridimensional, basta acrescentar mais uma equação de deslocamento que corresponde à coordenada z, conforme mostrado na Figura 7 e nas equações 3.7 e 3.8.

$$x_1 = x_0 + \Delta x$$
$$y_1 = y_0 + \Delta y$$
$$z_1 = z_0 + \Delta z \tag{3.7}$$

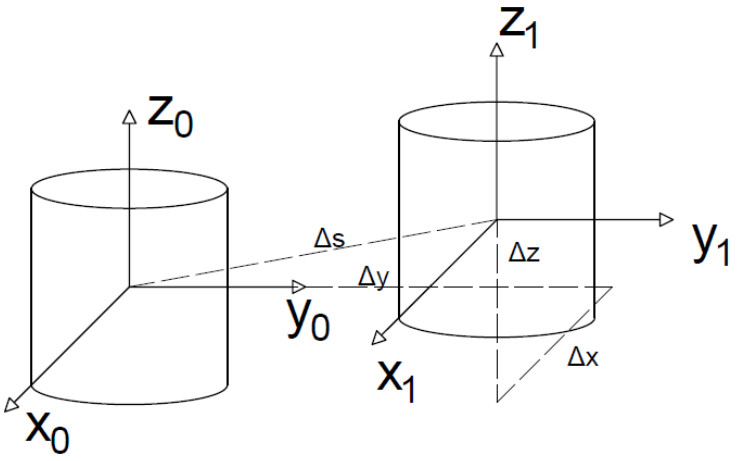

Figura 7 – Translação tridimensional de um corpo rígido

ou, em versão matricial,

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} \tag{3.8}$$

3.1.2 Rotações

As rotações são torções nos eixos coordenados de um sistema de referência (*frame*). Uma rotação em um dos eixos coordenados de um *frame* 0 produz um novo *frame* 1 no qual o ângulo deste eixo de rotação permanece inalterado, enquanto os outros eixos deslocam de um certo ângulo θ com relação ao *frame* 0. O sentido positivo dessa rotação segue a Regra de Fleming, também conhecida por Regra da Mão Direita, e a unidade do ângulo é dado em radianos ou graus. Exemplos de rotações são mostrados na Figura 8.

Conforme relatado anteriormente, um manipulador robótico é construído por juntas e *links*. As juntas são elementos ativos que podem realizar movimentos de translações e rotações. Uma junta rotacional é um elemento eletromecânico que realiza movimento de rotação em torno de seu próprio eixo de ação. Para modelar a cinemática deste aparato, as transformações homogêneas de rotação são excelentes alternativas.

A facilidade de se trabalhar com matrizes de rotação está em suas propriedades: as suas colunas representam os eixos coordenados no referencial móvel; as suas linhas representam os eixos coordenados no referencial fixos (da base, ou, inercial); a Norma de cada vetor é unitária, facilitando análise métrica no espaço cartesiano; seu determinante é sempre igual a um, ou seja, a transformação inversa é sempre possível; o produto interno de qualquer duas linhas ou qualquer duas colunas é igual a zero (os eixos coordenados são sempre ortogonais).

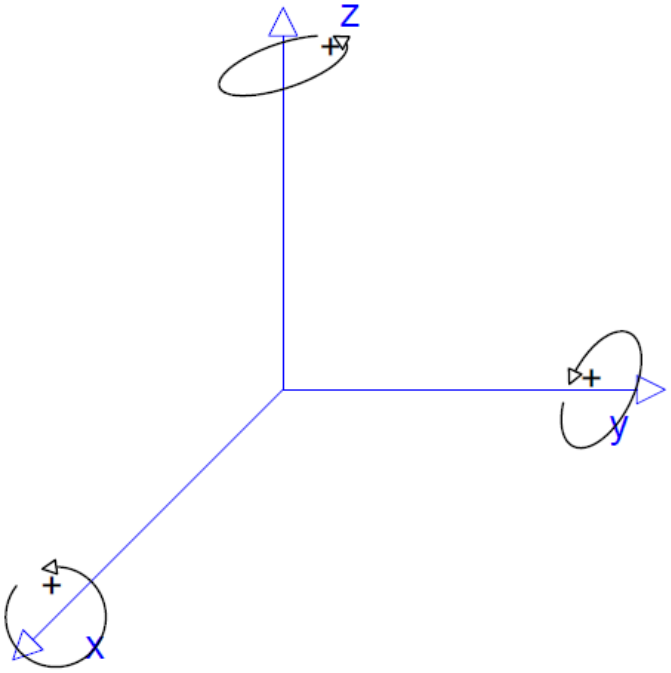

Figura 8 – Rotação dos eixos cartesianos.

Uma propriedade útil refere-se à inversa de uma matriz rotacional, que é igual a sua transposta. A rotação no eixo-z é conhecida com *yaw*. O seu eixo z_1 é mantido no mesmo lugar que o z_0 enquanto os eixos x_1 e y_1 sofrem uma rotação de θ com relação aos eixos x_0 e y_0 [23]. A Figura 9 evidencia o movimento de *yaw*.

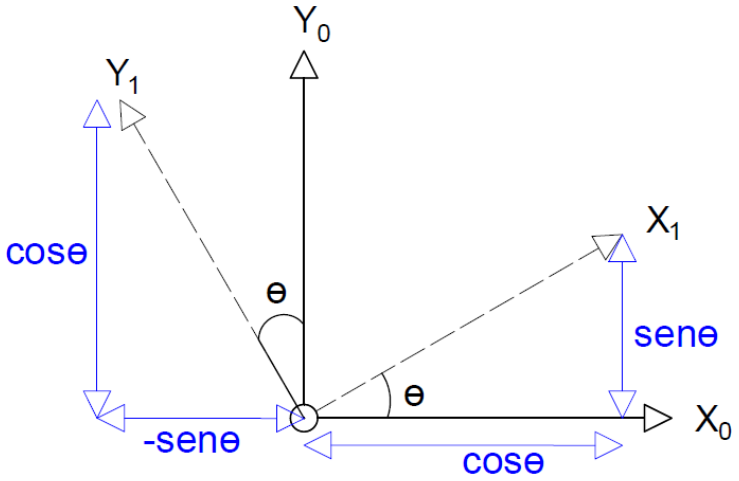

Figura 9 – Rotação no eixo-z (*yaw*).

Conforme a Figura 9 evidencia, o novo eixo-x com relação ao *frame* inercial {0} é mostrado na equação 3.9.

$$\begin{aligned} x_1^{x_0} &= cos\theta \\ x_1^{y_0} &= sen\theta \end{aligned} \tag{3.9}$$

O novo eixo-y com relação ao *frame* inercial é mostrado na equação 3.10.

$$\begin{aligned} y_1^{x_0} &= -sen\theta \\ y_1^{y_0} &= cos\theta \end{aligned} \tag{3.10}$$

Os novos eixos-*x,y,z*, do *frame* {1}, são dados com relação ao *frame* {0} inercial por meio das seguintes relações matriciais:

$$\mathbf{x_1} = \begin{bmatrix} cos\theta \\ sen\theta \end{bmatrix} \; e \; \mathbf{y_1} = \begin{bmatrix} -sin\theta \\ cos\theta \end{bmatrix} \tag{3.11}$$

As matrizes 3.11 descrevem apenas uma rotação com relação ao eixo-z. As únicas transformações que ocorrem são nos eixos-x e y. Porém, todo ponto representado do *frame* inercial deve ser representado no novo *frame* rotacionado {1}. Assim, a representação de um ponto qualquer no *frame* inercial como $P = [p_{x_0}, p_{y_0}]^T$ é dado no *frame* {1} como:

$$\begin{aligned} p_{x_1} &= p_{x_0} cos\theta - p_{y_0} sen\theta \\ p_{y_1} &= p_{x_0} sen\theta + p_{y_0} cos\theta \end{aligned} \tag{3.12}$$

Colocando essas equações em forma matricial:

$$\begin{bmatrix} p_x^1 \\ p_y^1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sen\theta \\ sen\theta & cos\theta \end{bmatrix} . \begin{bmatrix} p_x^0 \\ p_y^0 \end{bmatrix} \tag{3.13}$$

A matriz $\mathbf{R_z}(\theta)$ é chamada de matriz de rotação. Ela é responsável por fazer a rotação de um *frame* inicial para um *frame* final. No caso essa realiza uma rotação no eixo-z. A Figura 10 mostra uma rotação bidimensional em z de um ponto qualquer.

$$\mathbf{R_z}(\theta) = \begin{bmatrix} cos\theta & -sen\theta \\ sen\theta & cos\theta \end{bmatrix} \tag{3.14}$$

Em um espaço tridimensional, sabemos que o eixo-z não é alterado pela matriz de rotação. Logo basta escrever matricialmente.

$$\begin{bmatrix} p_x^1 \\ p_y^1 \\ p_z^1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sen\theta & 0 \\ sen\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} p_x^0 \\ p_y^0 \\ p_z^0 \end{bmatrix} \tag{3.15}$$

Se faz necessário escrever essa matriz de rotação no formato de coordenadas generalizadas para que possa ser utilizada em conjunto com as translações tridimensionais:

$$\begin{bmatrix} p_x^1 \\ p_y^1 \\ p_z^1 \\ 1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sen\theta & 0 & 0 \\ sen\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} p_x^0 \\ p_y^0 \\ p_z^0 \\ 1 \end{bmatrix} \tag{3.16}$$

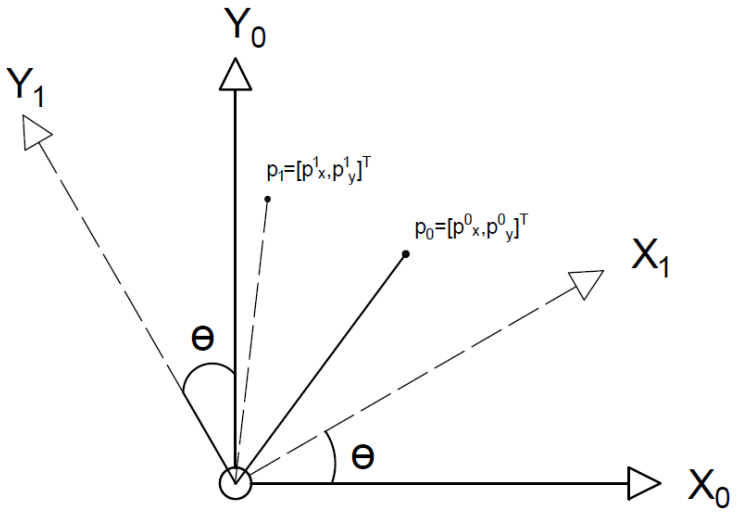

Figura 10 – Rotação no eixo-z definido um ponto.

A rotação no eixo-x, mostrada na Figura 11, é chamada de *Roll*, em que se rotaciona o eixo-x de um ângulo α, com equação cinemática exibida na Equação 3.17.

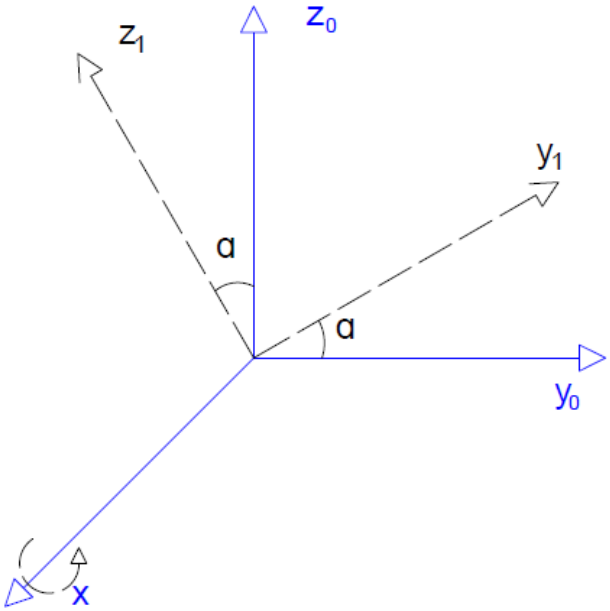

Figura 11 – *Roll*.

$$\begin{bmatrix} p_x^1 \\ p_y^1 \\ p_z^1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha & -sen\alpha & 0 \\ 0 & sen\alpha & cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} p_x^0 \\ p_y^0 \\ p_z^0 \\ 1 \end{bmatrix} \tag{3.17}$$

A rotação no eixo-y, mostrado na Figura 12, é conhecida como *Pitch*, em que o eixo-y é rotacionado de um angulo β, cuja cinemática é mostrada na Equação 3.18.

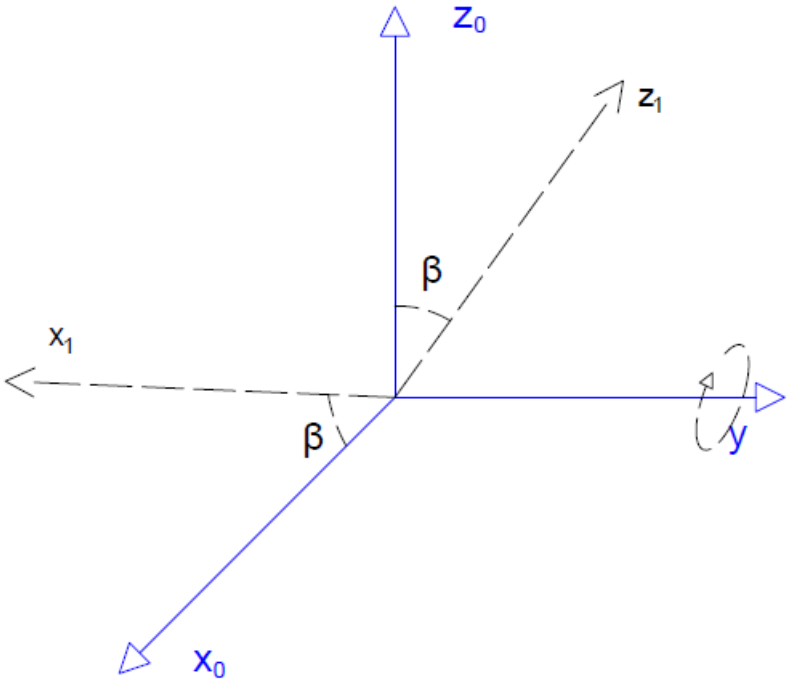

Figura 12 – *Pitch.*

$$\begin{bmatrix} p_x^1 \\ p_y^1 \\ p_z^1 \\ 1 \end{bmatrix} = \begin{bmatrix} cos\beta & 0 & sen\beta & 0 \\ 0 & 1 & 0 & 0 \\ -sen\beta & 0 & cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} p_x^0 \\ p_y^0 \\ p_z^0 \\ 1 \end{bmatrix} \tag{3.18}$$

3.2 Transformações Homogêneas

O manipulador robótico é, em termos cinemáticos, uma combinação de elos (*links*), juntas rotacionais e prismáticas. Com o domínio das matrizes de translação e rotação, a criação de qualquer cadeia cinemática se faz por meio de combinações entre elas. Esses processos de encadeamentos cinemáticos são chamados de transformações homogêneas. Geralmente as configurações das cadeias cinemáticas de um manipulador envolvem uma grande quantidade de rotações e translações. Para que a cadeia cinemática seja montada corretamente, as rotações e translações devem seguir as mesmas sequências de montagem do robô físico. A representação das cadeias cinemáticas é dada a seguir:

$$\mathbf{T_0^n} = \mathbf{T_0^1}.\mathbf{T_1^2}.\mathbf{T_2^3}...\mathbf{T_{n-1}^n} \tag{3.19}$$

A matriz resultante de todas as transformações é denominada de Matriz de Transformação Homogênea $\mathbf{T_0^n}$, que liga o *frame* {0} (base) ao frame final {n} (*end-effector*). Para que se alcance um resultado final desejado é importante respeitar a ordem das transformações conforme a montagem original do robô manipulador.

3.3 Modelagem por meio do método de Denavit-Hartenberg

O método de Denavit-Hartenberg é um método sistemático que se popularizou na robótica de manipuladores. Ele se tornou útil, pois facilitou calcular as cadeias cinemáticas de manipuladores com vários graus de liberdade. O método se tornou uma convenção entre os roboticistas para calcular cadeias cinemáticas complexas [22].

O método foi criado por Jacques Denavit e Richard Hartenberg. Jacques Denavit nasceu em Paris e fez sua graduação em engenharia mecânica na Northwestern University, Chicago, e sua área de interesse era a cinemática e dinâmica. Richard Hartenberg nasceu em Chicago, USA, fazendo sua graduação na University of Wisconsin-Madison, Wisconsin. Sua pesquisa na área da cinemática proporcionou uma nova base científica para aplicações em informática na análise de peças mecânicas complexas [22].

A convenção de Denavit-Hartenberg, mostrado na Figura 13, é representada pela sequência de quatro transformações, vistas em 3.20, sendo uma rotação no eixo-z (θ) seguida de uma translação no eixo-z (d), uma translação no eixo-x (a) e finalmente uma rotação no eixo-x (α).

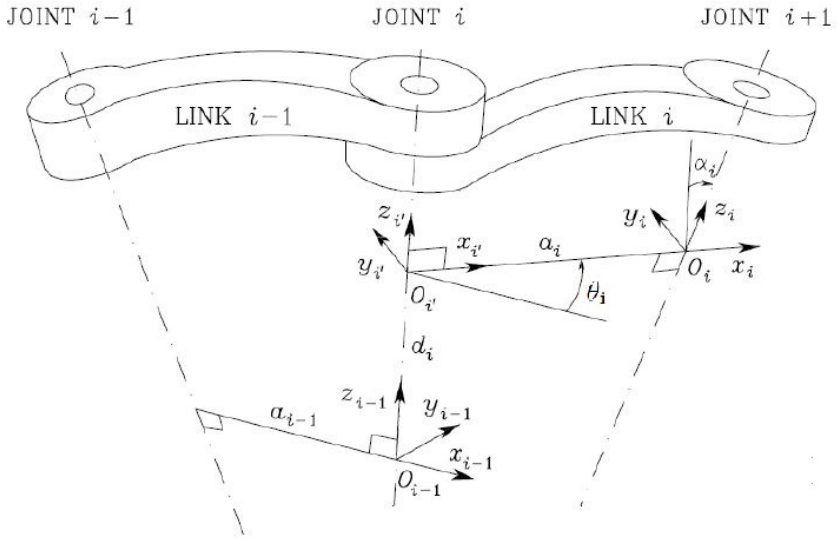

Figura 13 – Molelagem por meio Denavit-Hartenberg [2].

$$\mathbf{A_i} = \mathbf{R_z}(\theta_\mathbf{i})\mathbf{T_z}(\mathbf{d_i})\mathbf{T_x}(\mathbf{a_i})\mathbf{R_x}(\alpha_\mathbf{i}) \tag{3.20}$$

$$\mathbf{A_i} = \begin{bmatrix} cos(\theta_i) & -sen(\theta_i)cos(\alpha_i) & sen(\theta_i)sen(\alpha_i) & a_i cos(\theta_i) \\ sen(\theta_i) & cos(\theta_i)cos(\alpha_i) & -cos(\theta_i)sen(\alpha_i) & a_i sen(\theta_i) \\ 0 & sen(\alpha_i) & cos(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.21}$$

Há também a ordem inversa das transformações em outras fontes da literatura, resultando em uma matriz de transformação Denavit-Hartenberg diferente da apresentada em 3.21.

O algoritmo de transformações de Denavit-Hartenberg é bastante popular, sendo encontrado em quase todo livro-texto sobre robótica de manipuladores, uma vez que se tornou convenção para os roboticistas da área. Neste trabalho o algoritmo será utilizado para modelar um robô do tipo articulado, no qual as suas três juntas são rotacionais. A modelagem via Denavit-Hartenberg do robô em questão será apresentada posteriormente, em sessão específica sobre o robô.

3.4 Cinemática Inversa

Com a modelagem do manipulador, por Denavit-Hartenberg, a posição final da cadeia cinemática do robô é calculada a partir dos valores de cada variável de junta, em que esta abordagem é denominada por Cinemática Direta. No entanto, as aplicações úteis em problemas de engenharia necessitam do problema oposto, ou seja, a partir de posições e orientações desejadas, tem-se que determinar os parâmetros de juntas que as produzem.

Conforme descrito, robôs manipuladores, desde os mais simples até os mais complexos, possuem equacionamentos de cinemática direta não-lineares, nos quais tem-se como argumento os parâmetros de juntas, e obtém-se como resultado as posições e orientações dos robôs. Para resolver o problema inverso é necessário obter as equações inversas da cinemática direta, chamadas de Cinemática Inversa, do inglês *Inverse Kinematics, IK*.

A solução de tal problema pode ser feita por métodos analíticos para os casos em que os robôs possuem uma geometria simples ou então em casos bastante específicos, em que as equações da cinemática inversa podem ser encontradas. No entanto, se a complexidade da cadeia cinemática cresce, o cálculo das equações de cinemática inversa torna-se não-trivial ou até mesmo inviável em uma forma fechada.

Com o atual desenvolvimento da computação, a resolução desses problemas se torna possível em tempo real por meio de métodos numéricos iterativos, apesar desses serem normalmente de alta carga computacional. No entanto, pela inviabilidade de uma solução em forma fechada da cinemática inversa, faz-se necessário lançar mão de tais métodos.

Existem vários métodos iterativos usados para achar a cinemática inversa. Dentre eles tem-se: O Método de Newton-Raphson, Método da Inversa Generalizada de Moore-Penrose, Método do Jacobiano Transposto, Método do Gradiente Descendente, e assim por diante, sendo métodos de otimização numérica. A maioria desses métodos envolve a otimização de uma função objetivo por meio de retas tangentes, sendo estes derivados do método de Mínimos Quadrados [22]. Para este trabalho, a cinemática inversa em forma fechada do robô articulado não está disponível, portanto, a mesma será resolvida por meio

do método do Gradiente Descendente.

3.4.1 O Método do Gradiente Descendente

Para encontrar os parâmetros de junta que promovem a posição desejada para a ferramenta do manipulador se faz necessária a cinemática inversa. Quando uma forma fechada de equações não está disponível, é necessário utilizar um método numérico. Um dos métodos numéricos mais populares, não somente na resolução da cinemática inversa, mas para diversos problemas de otimização, seja em economia, mercados financeiros, física, química, biologia, dente outros, é o Método dos Gradiente Descendente.

O método, conforme o próprio nome sugere, faz a otimização de uma função objetivo por meio da ferramenta matemática do Gradiente. A função objetivo do método é uma função quadrática de erro, denominada por Erro Quadrático Médio, dada por:

$$f(P_k) = \left(\frac{G - P_k}{2}\right)^2 \tag{3.22}$$

em que $G = [G_x, G_y, G_z]^T$ é a posição objetivo tridimensional (*goal*) desejada para o manipulador, e $P_k = [P_x, P_y, P_z]_k^T$ é a posição atual do manipulador.

Conforme é possível perceber, a Equação 3.22 é uma parábola semidefinida positiva, em que seu ponto de mínimo é o ponto de mínimo erro entre o manipulador (P) e sua posição objetivo (G). De fato, no ponto de mínimo de uma parábola tem-se que a função objetivo é nula ($f(P_k) = 0$), denotando que o manipulador atingiu seu objetivo, ou seja, $P_k = G$. Para levar o erro até o ponto de mínimo o método faz uso do Gradiente.

No cálculo vetorial, o Gradiente de uma função $f(x)$, dado por $\nabla f(x)$, é um vetor que indica a direção da máxima variação da função $f(x)$. Tem-se que $f(x)$ é uma função de erro do tipo 3.22, e $\nabla f(x)$ informa, portanto, a direção de máxima variação do erro. Uma vez que se deseja encontrar o ponto de mínimo erro, ou seja, a direção de mínima variação do erro, é conveniente seguir na direção oposta à indicada pelo Gradiente, ou seja, $-\nabla f(x)$. Deste procedimento tem-se o nome do método, uma vez que a direção oposta ao Gradiente promove, a cada iteração, um comportamento decrescente do erro. O algoritmo é executado até que o erro $f(x)$ admita um valor suficientemente pequeno, $|f(x)| < \xi$, proporcionando a parada do algoritmo.

O método do Gradiente Descendente pode ser visto no Algoritmo 1.

while $e_k > \xi$ **do**
    $x_k = x_{k-1} - \eta \nabla f(x_{k-1})$;
    $e_k = |f(x_k)|$;
end

Algoritmo 1: Algoritmo do Gradiente Descendente

em que e é o erro quadrático médio e η é um fator de escala conhecido como taxa de aprendizado, utilizada para evitar que o algoritmo seja atualizado a passos muito grandes (dependendo do valor do Gradiente), os quais, se ocorrerem, podem inviabilizar a solução e até mesmo instabilizar o algoritmo.

Desta maneira, o Gradiente Descendente será utilizado para resolver a cinemática inversa comandada pelo usuário do manipulador, de modo a proporcionar ao robô alcançar a posição desejada, tornando-o uma ferramenta útil para o usuário portador de dificuldades de locomoção.

3.5 Campos Potenciais

O algoritmo de campos potencias foi criado para gerar Caminhos capazes de evitar obstáculos. É um método bastante popular em aplicações industriais e acadêmicas. O algoritmo se comporta tanto como um planejador de caminhos como um controlador [23, 22]. Este trabalho utilizará o algoritmo de campos potenciais de maneira a auxiliar o usuário do robô manipulador a cumprir o objetivo evitando a colisão do robô com o ambiente, como controle compartilhado (*shared control*).

Os campos potenciais se baseiam no comportamento de partículas com cargas elétricas sobre o efeito de um campo potencial. O método foi criado por Andrews e Hogan em 1983 [24], para o controle e proteção contra colisões de robôs manipuladores, conhecido pelo seu termo em inglês *Pontetial Fields*. O método foi adaptado para a robótica móvel em 1985, por Khatib [25] e, desde então, se popularizou na literatura por sua simplicidade e efetividade na presença de obstáculos convexos.

A estrutura do algoritmo reside na analogia com forças atrativas e repulsivas entre cargas elétricas carregadas. No caso deste método, o efetuador do robô P_k (*end-effector*) será representado por uma carga positiva e o ponto objetivo G por uma carga negativa, para que exista uma força de atração F_{att} entre ambos de modo que o efetuador seja levado ao objetivo por essa força F_{att}. Já os obstáculos serão representados por cargas positivas para que exista uma força de repulsão F_{rep} entre estes e o robô. Dessa maneira, a força total (ou, resultante) F_{tot} da combinação das forças F_{att} e F_{rep} possui a propriedade de, ao mesmo tempo, levar o robô ao ponto objetivo desviando dos obstáculos do ambiente. As forças de atração e repulsão são criadas por campos Potenciais (U), os quais são definidos em 3.23:

$$U(q) : R^n \rightarrow R \tag{3.23}$$

A modelagem do campo atrativo U_{att} é dado por uma quadrática de acordo com o exposto em [24, 25]. O campo atrativo é mostrado na equação 3.24:

$$U_{att} = \frac{1}{2} K_{att} \rho^2 \tag{3.24}$$

em que $\rho = \sqrt{(x_G - x_R)^2 + (y_G - y_R)^2}$ é a distância Euclidiana entre o efetuador do robô (x_R, y_R) e o ponto objetivo (x_G, y_G), e K_{att} é uma constante positiva denominada por constante atrativa, a ser calibrada para cada aplicação. Nota-se que o campo atrativo é centrado no ponto objetivo e decresce com o quadrado da distância ρ, tornando-se cada vez menor conforme o efetuador se aproxima do objetivo. O campo atrativo é exibido na Figura 14.

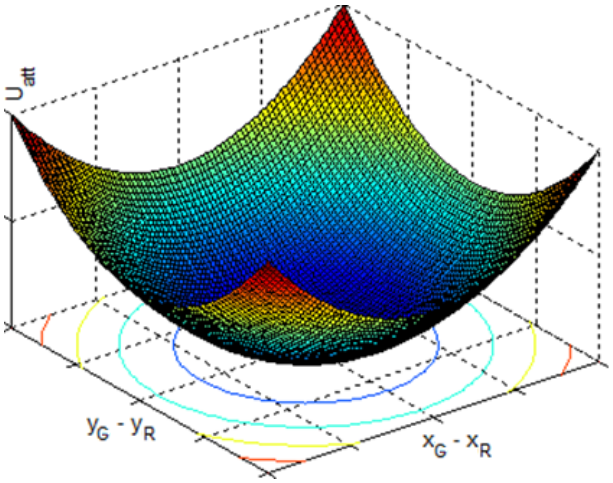

Figura 14 – Campo potencial atrativo U_{att} [2].

A força atrativa F_{att} é obtida uma vez que um objeto qualquer (no caso, o efetuador do manipulador) adentra o campo. A força atrativa age sobre o corpo de forma a impulsioná-lo na direção do menor potencial existente no campo, o ponto objetivo. É dada, portanto, pela direção oposta à do Gradiente do campo, desta maneira, mostrada em 3.25.

$$F_{att} = -\nabla(U_{att}) \tag{3.25}$$

Aplicando a Equação 3.24 na 3.25, tem-se em 3.26 a força atrativa entre o efetuador do manipulador e o ponto objetivo [2]:

$$F_{att} = K_{att} \begin{bmatrix} x_G - x_R \\ y_G - y_R \end{bmatrix} \tag{3.26}$$

Uma das vantagens do algoritmo dos campos potenciais é a possibilidade de desviar de obstáculos. Para tanto, é necessário um campo repulsivo de modo a gerar uma força repulsiva que impulsiona o robô para longe dos obstáculos. Esse campo repulsivo deve ter a mesma carga que o efetuador do manipulador (positiva), e sua intensidade deve aumentar ao se aproximar do obstáculo. A modelagem é mostrada em 3.27:

$$U_{rep} = \begin{cases} \frac{1}{2} K_{rep} \left(\frac{1}{\epsilon} - \frac{1}{\epsilon_0} \right)^2, & \epsilon \leq \epsilon_0 \\ 0, & \epsilon > \epsilon_0 \end{cases} \tag{3.27}$$

em que $\epsilon = \sqrt{(x_O - x_R)^2 + (y_O - y_R)^2}$ é a distância Euclidiana entre um obstáculo (x_O, y_O) e o robô (x_R, y_R), ϵ_0 é uma distância de horizonte de eventos, a partir da qual o campo repulsivo começará a agir sobre o efetuador do robô, e K_{rep} é uma constante positiva, denominada constante repulsiva, a ser calibrada para cada aplicação. O campo repulsivo é representado na Figura 15.

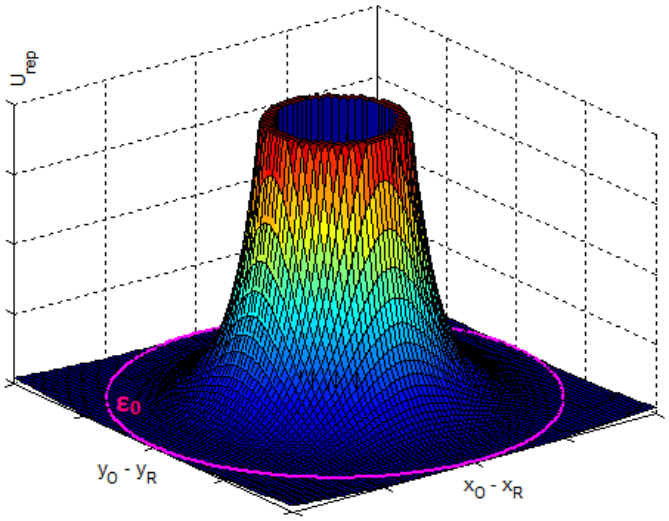

Figura 15 – Campo potencial repulsivo U_{rep} [2].

A força repulsiva é obtida como a atrativa, pela direção oposta à do Gradiente:

$$F_{rep} = -\nabla(U_{rep}) \tag{3.28}$$

Quando a Equação 3.27 é aplicada em 3.28 obtém-se:

$$F_{rep} = \frac{K_{rep}}{\epsilon^3}\left(\frac{1}{\epsilon_0} - \frac{1}{\epsilon}\right)\begin{bmatrix} x_O - x_R \\ y_O - y_R \end{bmatrix} \tag{3.29}$$

Presume-se que existam vários obstáculos no ambiente onde o robô está inserido, desta maneira, não existe uma única força repulsiva atuando sobre o robô. Logo é necessário realizar o somatório das forças repulsivas geradas por todos os obstáculos dentro do horizonte de eventos ϵ_0. Os obstáculos são detectados por sensores capazes de dicerní-los, tais qual sensores *rangefinders* como os SONARES e LIDARES, bem como câmeras munidas de *software* de identificação de objetos, dos quais é possível abstrair as distâncias Euclidianas requeridas pela F_{rep}. Por conseguinte, a força repulsiva resultante da iteração de todos os N_0 obstáculos inseridos em ϵ_0 é mostrada em 3.30:

$$F_{rep} = K_{rep}\sum_{i=1}^{N_0}\frac{1}{\epsilon_i^3}\left(\frac{1}{\epsilon_0} - \frac{1}{\epsilon_i}\right)\begin{bmatrix} x_0^i - x_R \\ y_0^i - y_R \end{bmatrix} \tag{3.30}$$

O somatório de F_{att} e F_{rep}, evidenciada em 3.31 é capaz de levar o robô até o objetivo com segurança. A Figura 16 representa o campo resultante.

$$F_{tot} = F_{att} + F_{rep} \tag{3.31}$$

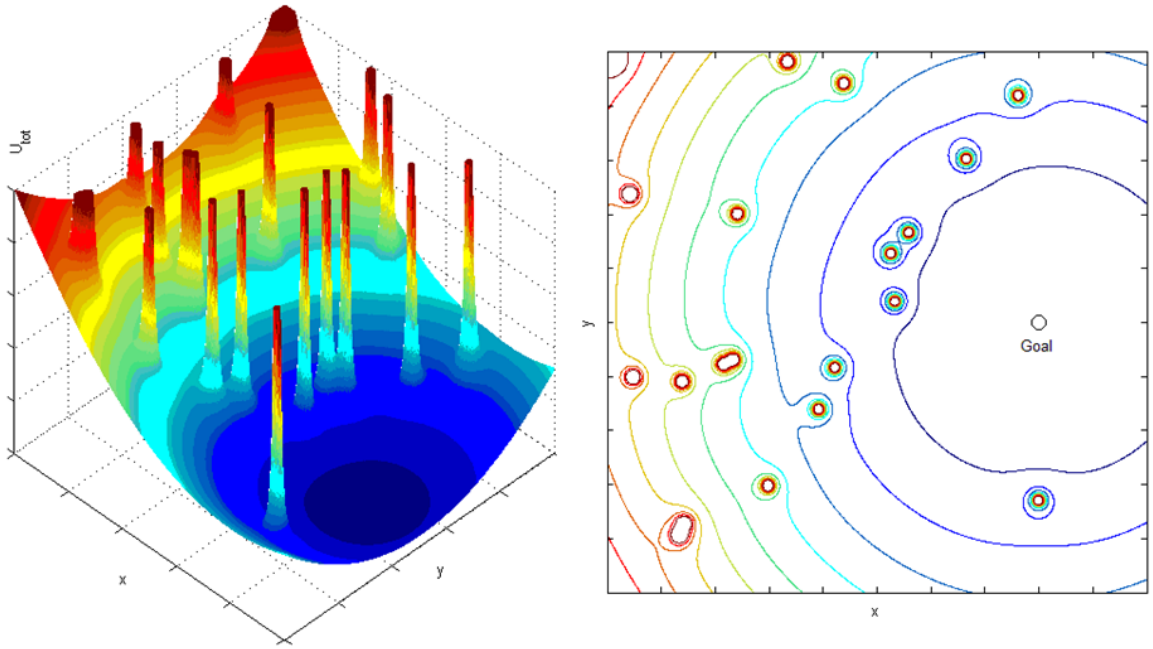

Figura 16 – Campo potencial repulsivo U_{tot} [2].

3.6 Robô montado para o trabalho

O robô que foi montado nesse trabalho foi inspirado no modelo ABB-IRB 1410 [3], mostrado na Figura 17. O robô mencionado é do tipo articulado com 3 graus de liberdade, em que todas as suas juntas são rotacionais. A escolha desse modelo se deu pela sua configuração antropomórfica, facilitando a assimilação de controle por parte do usuário utilizando controle compartilhado, e por oferecer um amplo espaço de trabalho.

Os parâmetros de Denavit-Hartenberg para o robô construído foram obtidos deste modelo ABB pelo método da seção 3.3 e estão mostrados na Tabela 1. Com esse parâmetros, a matriz 3.33 de transformação é calculada e exibida na Equação 3.33.

Os parâmetros L_1, L_2 e L_3 correspondem aos tamanhos dos Links do manipulador. Em que a distância entre as duas primeiras juntas equivale ao primeiro link, a distância entre a segunda e a terceira juntas é análogo ao segundo link e a distância entre a ferramenta de trabalho e a terceira junta é igual ao terceiro link. Os θ_1^* *, θ_2^* e θ_3^* representam a variação dos ângulos de suas juntas com relação ao eixo-z.

Tabela 1 – Tabela Transformações Denavit-Hartenberg Modelo ABB [3].

Link	**d**	$\boldsymbol{\theta}$	**a**	$\boldsymbol{\alpha}$
1	L_1	θ_1^*	0	90°
2	0	θ_2^*	L_2	0
3	0	θ_3^*	L_3	0

$$\mathbf{T_0^3 = A_1.A_2.A_3} \tag{3.32}$$

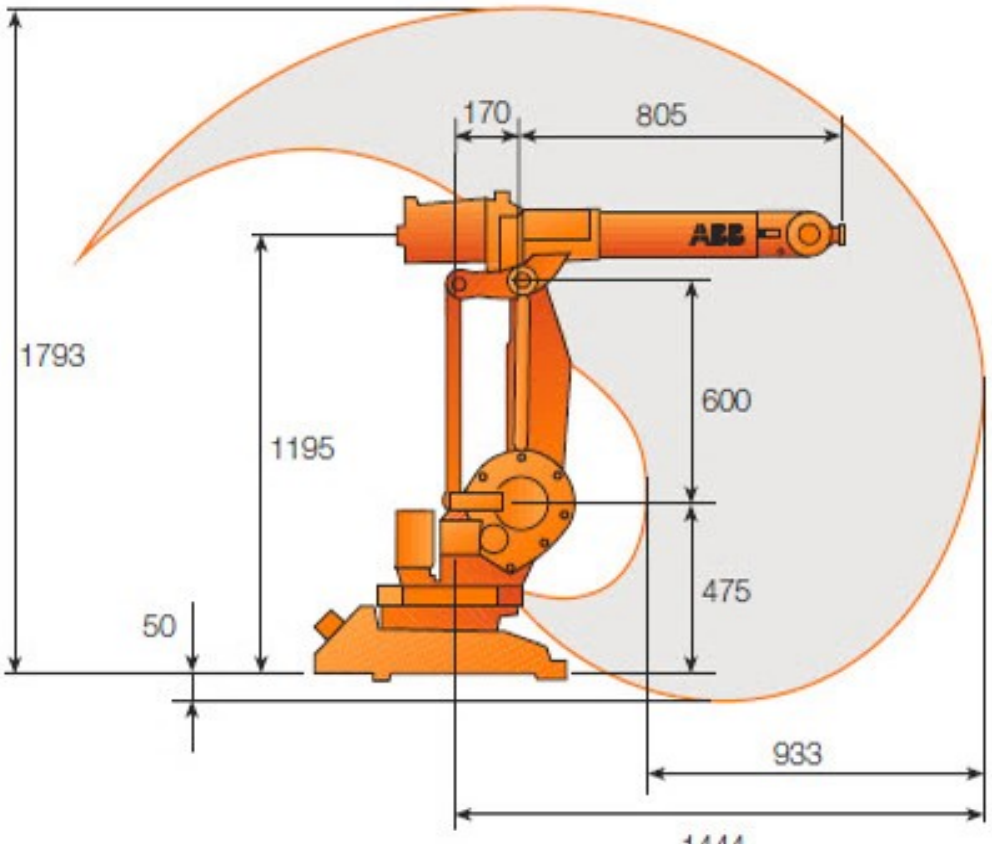

Figura 17 – ABB-IRB 1410 [3]

$$\mathbf{T_0^3} = \begin{bmatrix} c_{\theta_2+\theta_3}.c_{\theta_1} & -s_{\theta_2+\theta_3}.c_{\theta_1} & s_{\theta_1} & c_{\theta_1}.(L_3.c_{\theta_2+\theta_3} + L_2.c_{\theta_2}) \\ c_{\theta_2+\theta_3}.s_{\theta_1} & -s_{\theta_2+\theta_3}.s_{\theta_1} & -c_{\theta_1} & s_{\theta_1}.(L_3.c_{\theta_2+\theta_3} + L_2.c_{\theta_2}) \\ s_{\theta_2+\theta_3} & c_{\theta_2+\theta_3} & 0 & L_1 + L_3.s_{\theta_2+\theta_3} + L_2.s_{\theta_2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.33}$$

A matriz $\mathbf{T_0^3}$ calculada através da multiplicação das matrizes de transformação obtidas pelo método de Denavit-Hartengerb, na seção 3.3, mostrado na equação 3.32. Uma observação na matriz 3.33 é com relação a sua representação dos seus termos. Em que $c_{\theta_n+\theta_m}$ é equivalente a $cos(\theta_n + \theta_m)$, $s_{\theta_n+\theta_m}$ é igual a $sen(\theta_n + \theta_m)$, c_{θ_n} é análogo a $cos(\theta_n)$ e s_{θ_n} é similar a $sen(\theta_n)$. Representando a matriz final de transformação.

Uma vez com a modelagem cinemática disponível, é possível detalhar a construção do robô articulado deste trabalho. A seguir seguem os detalhes e materiais utilizados.

3.6.1 Material Utilizado na construção do robô manipulador

- Três Servos Motores;
- Quatro Links de Plástico;
- Uma caixa de Madeira;

- Arduino® Nano V3.0;
- Chave on-off;

3.6.2 Arduino® Nano V3.0

O Arduino® Nano na Figura 18 é pequeno, completo e oferece uma arquitetura amigável baseado no microcontrolador ATmega328 (Arduino® 3.X) ou no ATmega168 (Arduino® Nano 2.X). No caso deste trabalho, escolheu-se trabalhar com as suas portas digitais I/O. O Arduino utilizado oferece 14 pinos dos quais seis desses possuem saída de sinal de largura de pulso (PWM). No qual utiliza 3 saídas PWM para controlar os servos motores [4].

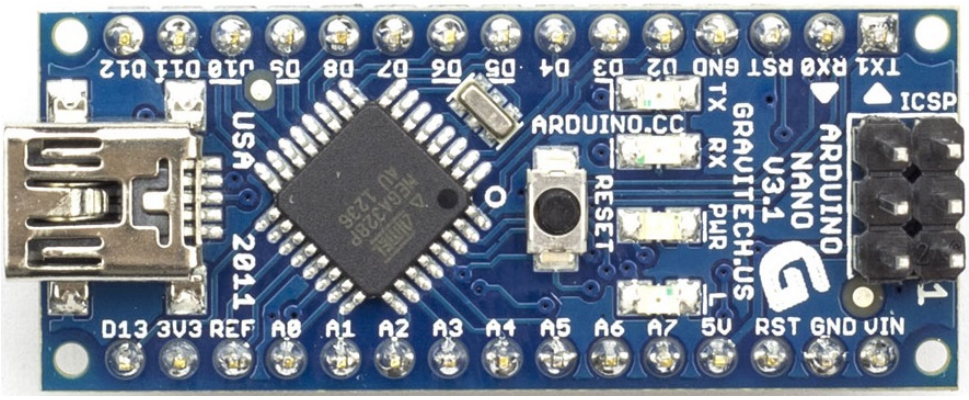

Figura 18 – Arduino® Nano [4]

O Arduino® oferece facilidades no estabelecimento da comunicação com o computador, oferecendo comunicação serial UART TTL (5V) que estão disponíveis nos pinos digitais 0 (RX) e 1(TX). Além disso, oferece comunicação serial via USB com os Drivers FTDI (disponível no Hardware do Arduino®), o qual fornece uma porta COM virtual para o *software* desenvolvido no computador. Ele possui um monitor serial que permite a visualização dos dados enviados para o processador. Este processador é responsável por receber os valores angulares calculados pelo algoritmo Gradiente Descendente e realizar a modulação PWM, referente a tais ângulos, para acionar os servos motores, que correspondem às juntas do manipulador.

3.6.3 Servomotores

Um servomotor é uma máquina eletromecânica capaz de realizar controle de posição e velocidade por meio de um sistema de controle em malha fechada. No encapsulamento de servo motor existe um sensor de cálculo de posições ou velocidades angulares. Porém, cada fabricante usa um tipo de sensor deferente, em que, na maioria dos casos trata-se de um potenciômetro acoplado ao motor do servo. O sistema realiza uma constante leitura da posição do motor, podendo assim ser comparado com uma referência. Internamente

existe também um controlador, normalmente um PID, que atua para minimizar o erro entre a leitura do sensor e a referência.

A referência de posição é dada por um sinal de largura de pulso (PWM), no qual é definido uma taxa de repetição, com largura mínima e máxima. O servo HS-422, mostrado na Figura 19, trabalha com um controlador e tem sua posição de neutro em 1500us. Os detalhes estão mostrados na Figura 20.

Figura 19 – Servomotor HS422 [5]

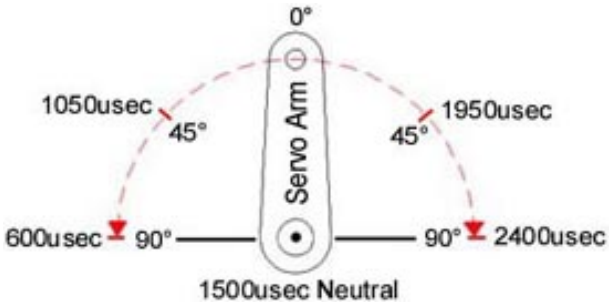

Figura 20 – Esquema da Posição dos Ângulos [5]

O servo pode ser alimentado com uma tensão entre 4,8 ate 6,0 volts. Se alimentado com 6,0 V possui um torque de 4,1 kg.cm, e a posição de seus ângulos pode variar de 0 a 180°. Esse servo será usado para compor as juntas rotacionais do manipulador.

3.6.4 Comunicação

Toda a cinemática para o manipulador é calculada por um computador externo, que está ligado diretamente com a interface assistiva do usuário. Uma vez que o usuário emita um comando para o manipulador, os algoritmos do gradiente descendente e dos campos potenciais, os quais foram implementados em MATLAB®, entram em ação para que o robô execute o comando de maneira acurada e segura. Assim, os programas de controle conseguem calcular os ângulos de cada uma das juntas para que o efetuador

alcance a posição desejada. Uma vez obtidos, esses ângulos das juntas são enviados de maneira serial para o Arduino®.

Existe um pacote de suporte em MATLAB® [20] para Arduino® [21]. Esse pacote permite usar o Arduino® conectado ao computador do usuário e executar suas entradas, saídas digitais e analógicas. Primeiramente é feito um *upload* de um programa servidor no microcontrolador, que é responsável por receber constantemente os comandos que chegam pelo MATLAB® via porta serial. Recebendo um comando, o servidor executa a ação e retorna um resultado. Esse servidor pode ser encontrado no sítio da Mathworks [20]. Essa ferramenta é útil pois toda programação é feita em MATLAB®, com todas as facilidades oferecidas por esse pacote computacional, como os *plots* (gerador de figuras), as ferramentas de *debug*, muitas funções prontas e a linguagem de rápido aprendizado.

3.6.5 Circuito Impresso

Para realizar as conexões entre a fonte de alimentação, os servomotores e o Arduino® foi necessário fabricar uma placa de circuito impresso, mostrada na Figura 21. Nessa placa o circuito de alimentação dos motores foi separado do circuito de comando do Arduino® para a proteção do mesmo. Além disso, também foi instalada uma chave ON/OFF para desligar de maneira direta a alimentação dos servomotores quando necessário, para preservá-los. Foram usadas as saídas D3, D10 e D11 do Arduino® uma vez que são as saídas que fornecem o sinal PWM. A alimentação do Arduino® ocorre pelo cabo USB conectado ao computador do usuário.

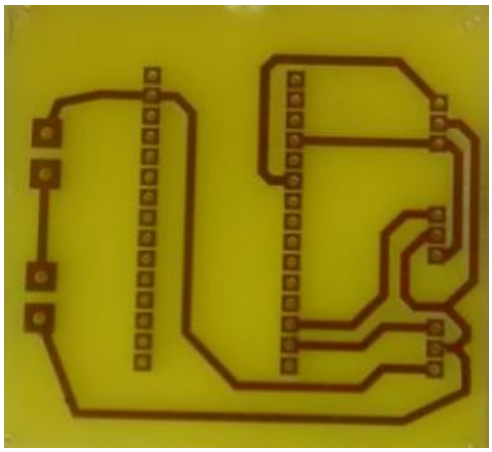

Figura 21 – Placa de Circuito Impresso.

3.6.6 Montagem do robô

O manipulador confeccionado para esse trabalho é um robô articulado RRR (três juntas Rotacionais), que é conhecido também como robô articulado ou antropomórfico. As suas juntas são servomotores HS-422 e seus elos são formados por extensões de plástico e estruturas metálicas como podemos ver na Figura 22, aqui repetida a partir do Capítulo 1.

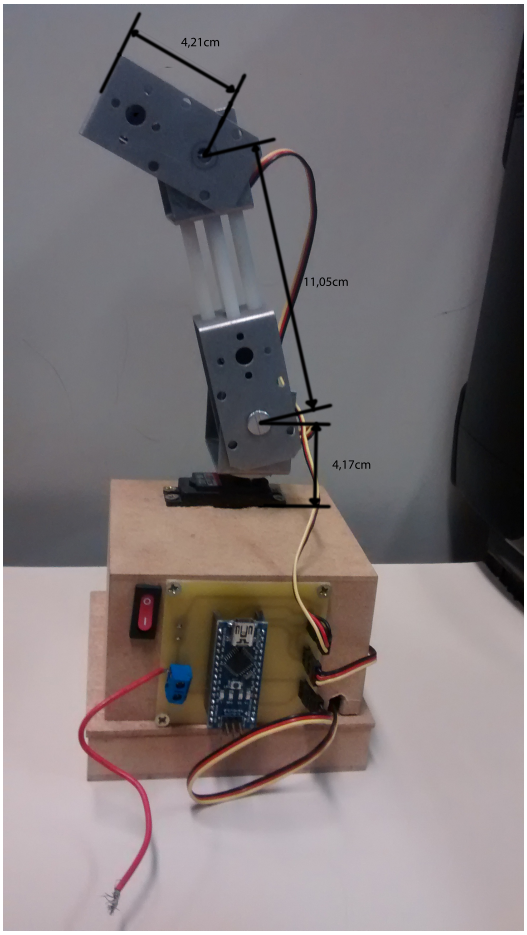

Figura 22 – Robô montado de 3 graus de liberdade.

Uma comparação do robô feito nesse trabalho com o modelo ABB- IRB 1410 [3] é mostrado na Figura 23. Embora as dimensões do robô sejam menores, a configuração das três juntas rotacionais são as mesmas. Os tamanhos dos elos do robô estão na Tabela 2, a partir dos quais pode-se fazer a cinemática direta por meio do método de Denavit-Hartenberg (Seção 3.3). Para calcular a função correspondente a posição espacial de cada junta.

Com a cinemática direta calcula a posição espacial de cada uma das juntas e do efetuador do robô dado seus parâmetros de juntas. No qual são mostradas nas equações 3.34, 3.35 e 3.36.

Tabela 2 – Tamanho dos elos do robô.

L_1	L_2	L_3
$4.17cm$	$11.05cm$	$4.21cm$

$$Junta_{1,2} = \begin{bmatrix} 0 \\ 0 \\ L_1 \end{bmatrix} \tag{3.34}$$

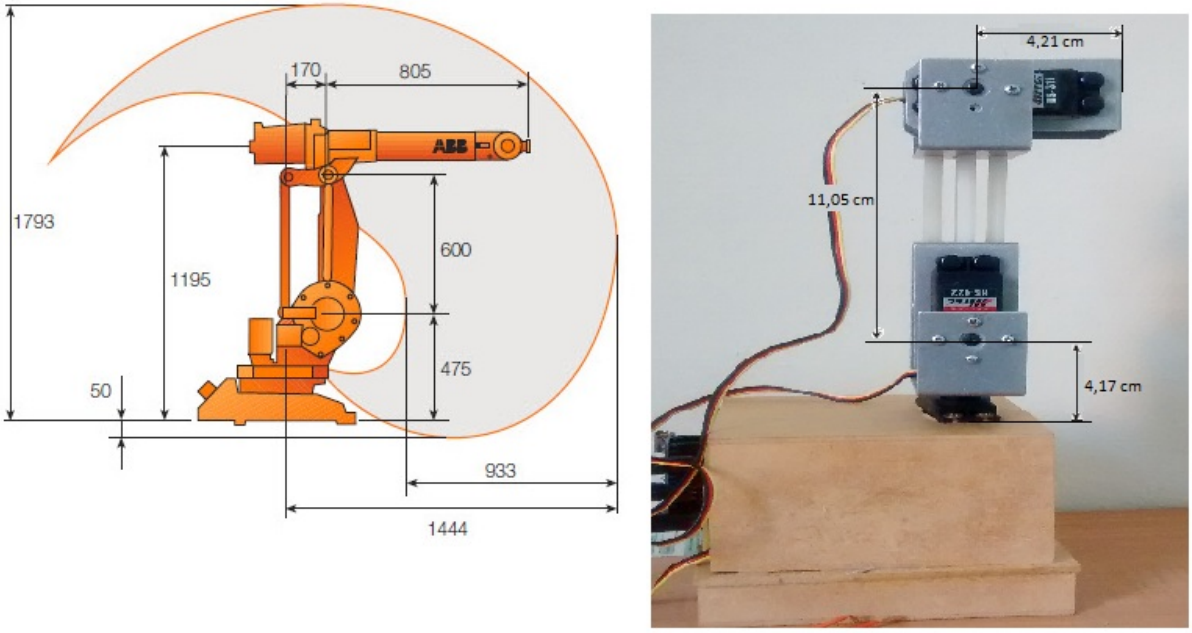

Figura 23 – Comparação com o ABB - IRB 1410 [3].

$$Junta_3 = \begin{bmatrix} L_2 cos(\theta_1).cos(\theta_2) \\ L_2 cos(\theta_2).sen(\theta_1) \\ L_1 + L_2.sen(\theta_2) \end{bmatrix} \tag{3.35}$$

$$Efetuador = \begin{bmatrix} cos(\theta_1).[L_3.cos(\theta_2 + \theta_3) + L_2.cos(\theta_2)] \\ sen(\theta_1).[L_3.cos(\theta_2 + \theta_3) + L_2.cos(\theta_2)] \\ L_1 + L_3.sen(\theta_2 + \theta_3) + L_2.sen(\theta_2) \end{bmatrix} \tag{3.36}$$

Uma vez determinadas as equações da cinemática direta do robô, é possível calcular a posição espacial do efetuador com relação ao sistema de coordenadas inercial (base do robô). Dessa maneira, além do robô real, é possível simular o robô, o que possibilita a calibração de parâmetros e testes computacionais antes da utilização do equipamento real.

Com isso, tem-se todas as ferramentas necessárias para aplicar um controle de posição com um planejador de caminhos de modo a obedecer tarefas específicas comandadas por um usuário. O próximo capítulo mostrará a conexão de todas essas ferramentas computacionais integradas com o robô simulado e o real, para que comandos dados por um usuário sejam obedecidos com precisão, acurácia e segurança para o usuário e para o equipamento.

4 Resultados

Esse capítulo se propõe a validar a estratégia proposta ao longo do texto, por meio de uma série de testes simulacionais e práticos. O conjunto integrado das diversas técnicas descritas deverão, em última análise, ser capaz de comandar o robô manipulador com fidelidade e segurança conforme a operação designada pelo usuário utilizando a interface humano-máquina assistiva. A seguir tem-se as seções mostrando os diversos testes feitos e seus resultados.

4.1 Tempos computacionais

Embora o foco do trabalho não seja na otimização do tempo computacional, este é um parâmetro importante para verificação da plausibilidade de utilização dos algoritmos em tempo real, ou seja, com relação à execução de comandos do usuário. Testes com relação à cinemática inversa e ao algoritmo de planejamento foram executados e serão mostrados à seguir.

4.1.1 Tempo computacional da cinemática inversa

O algoritmo do Gradiente Descendente, mostrado na seção 3.4, foi implementado numa função em MATLAB®. Sabe-se que métodos de cinemática inversa iterativos são, usualmente lentos em termos de tempo de convergência. Para realizar o teste de tempo computacional desse algoritmo, definiu-se uma mesma condição inicial para o mesmo destino, sendo este no extremo oposto à posição inicial, ou seja, o local mais longe possível para o manipulador alcançar.

Foram coletadas 100 amostras de tempo relativos à convergência deste algoritmo. O resultado qualitativo é mostrado no gráfico de barras da Figura 24 e os resultados quantitativos estatísticos estão na Tabela 3. Conforme os resultados apontam, o algoritmo obteve tempos computacionais pequenos, com média de 2,08 milissegundos e com desvio padrão pequeno, de 0,03 milissegundos, se mostrando um algoritmo com desempenho rápido e consistente, passível de aplicações reais para usuários que necessitem desse tipo de aparelho assistivo.

Tabela 3 – Tempo Computacional Gradiente Descendente.

Média(ms)	**Desvio Padrão(ms)**
2.08	0.03

Este experimento foi feito por meio de simulação computacional, e diversos outros pontos foram testados, obtendo tempos consistentes com os resultados evidenciados nesta sessão, de maneira que o algoritmo do Gradiente Descendente sempre obteve os parâmetros

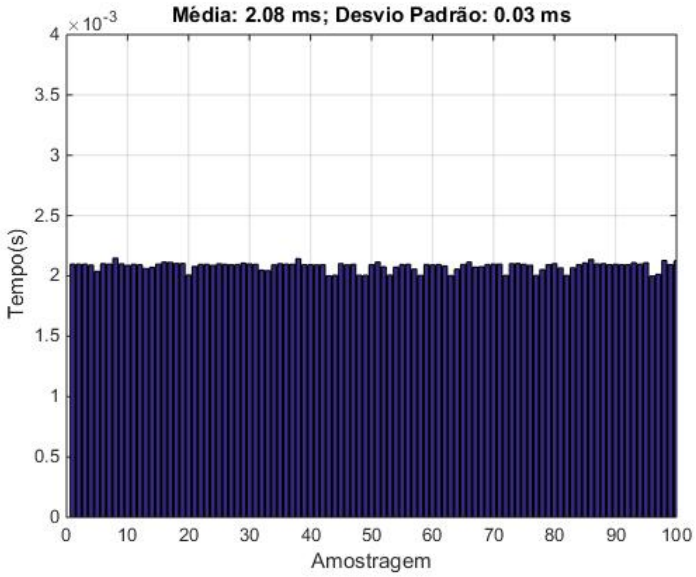

Figura 24 – Amostragem de Tempo Computacional do Algoritmo de Gradiente Descendente.

das juntas do robô de forma a cumprir a tarefa. A Figura 25 evidencia um teste para 60° em todas as juntas convergindo em 39 iterações do Gradiente Descendente.

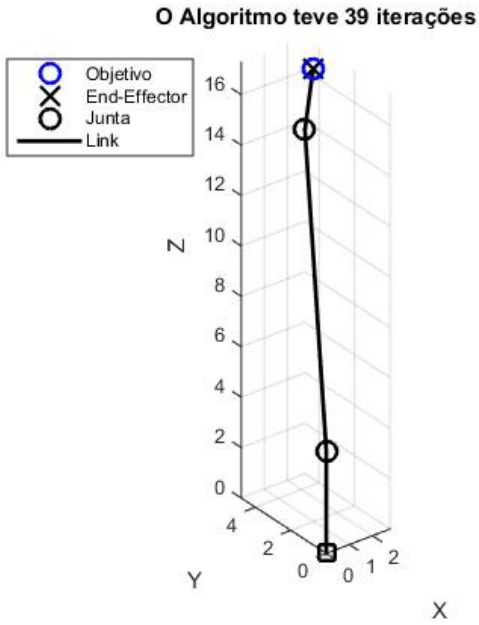

Figura 25 – Simulação do Robô usando Gradiente Descendente.

A razão do algoritmo de Gradiente Descendente sempre convergir para todas as posições desejadas é que, para o cálculo da cinemática inversa de posição, isto é, sem se preocupar com a orientação do efetuador, não existem mínimos locais na função objetivo de minimização (Equação 3.22, página 33). Uma vez que não existem mínimos locais, basta que a posição objetivo esteja dentro do espaço de trabalho (*workspace*) do robô que a mesma será alcançada.

4.1.2 Tempo computacional dos campo potenciais

A função que calcula os campos potenciais foi implementada em MATLAB®, e recebe os parâmetros de juntas atuais do manipulador, a posição objetivo e os obstáculos que estão no ambiente. Assim, a função calcula uma trajetória e retornará todas as posições que o efetuador realizou e seus respectivos ângulos de juntas. Essa função foi feita usando o algoritmo descrito na Sessão 3.5.

O primeiro teste se deu para determinar o tempo computacional desse método. Definiu-se uma posição objetivo dois centímetros abaixo da posição atual do robô (a qual pode ser qualquer) no eixo-z. Para este teste inicial, o ambiente não possui obstáculos, de modo que somente a força atrativa leve o efetuador ao objetivo em linha reta, o menor caminho possível. Com esses parâmetros, a função campos potenciais foi executada por 100 iterações, com seu tempo computacional coletado. O gráfico de barras é mostrado na Figura 26 e sua média e desvio padrão estão na Tabela 4.

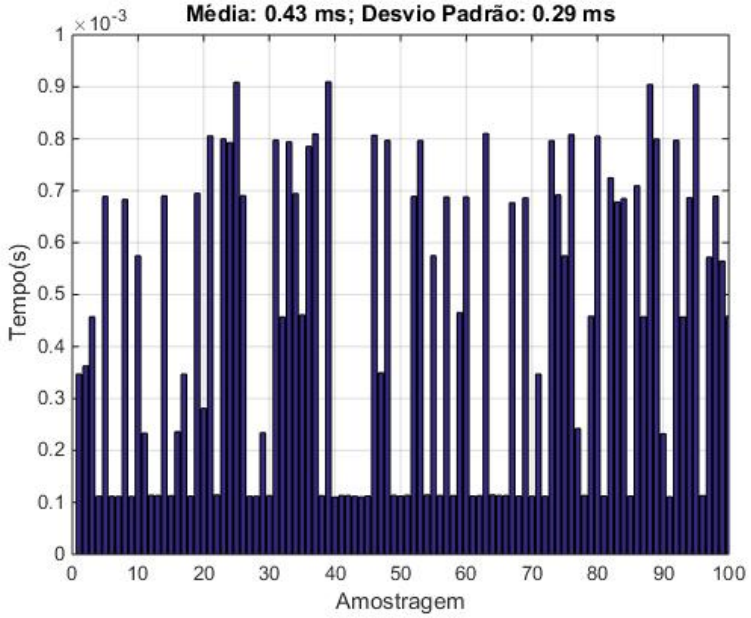

Figura 26 – Amostragem de tempo Computacional do Algoritmo de Gradiente Descendente.

Tabela 4 – Tempo Computacional Campos Potenciais.

Média(ms)	**Desvio Padrão(ms)**
0.43	0.29

Os resultados mostram que, apesar do tempo médio ser baixo, o desvio padrão foi relativamente alto, o que pode ser observado na Figura 26. A razão dessa variação se dá pelo fato de que algumas posições objetivo imprimem maiores dificuldades de serem alcançadas em razão das configurações assumidas pelo manipulador, pois várias delas estiveram próximas dos limites do espaço de trabalho, impondo dificuldades geométricas.

Ainda assim, o tempo médio mostrou-se baixo, sendo também compatível com aplicações em tempo real para usuários.

4.2 Restrições dos servomotores

A sessão 3.6 mostra que os servos motores usados no robô real possuem restrições de movimentação angular dentro do intervalo 0 a 180°. Essas restrições devem ser levadas em consideração no modelo cinemático do robô virtual, uma vez que o algoritmo do Gradiente Descendente (Sessão 3.4) retorna valores angulares quaisquer. Dessa maneira, com as restrições dos servos inseridas a função da cinemática inversa não retorna valores fora dos limites físicos das juntas, reduzindo, portanto o espaço de trabalho do robô. Quando uma posição desejada está fora do espaço de trabalho o algoritmo retorna a mensagem "posição inválida".

Como o espaço de trabalho do robô ficou bastante reduzido, diversas posições deixaram de ser alcançadas. Para contornar este problema, sempre que uma posição inválida é dada como objetivo, lançam-se diversas posições aleatórias ao seu redor. Essas posições aleatórias são testadas de forma a obter uma posição válida próxima da posição desejada. Uma vez que uma posição válida seja encontrada o robô a ela é direcionado. A Figura 27 mostra a situação de uma posição desejada fora do espaço de trabalho. Tem-se diversas tentativas e a posição válida é representada pelo “x” vermelho.

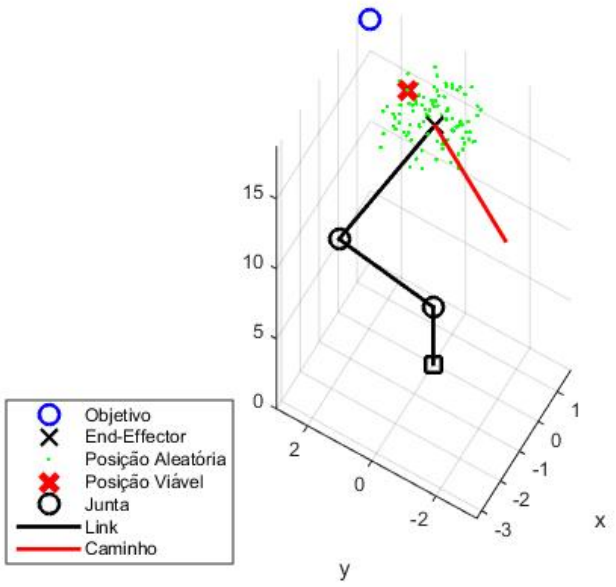

Figura 27 – A procura por pontos que atendam as restrições físicas do robô.

Ainda assim, é possível que nenhuma das diversas posições de tentativa sejam válidas, dessa maneira, retorna-se uma mensagem final de impossibilidade do cumprimento da tarefa, e o comando dado é cancelado, oferecendo ao usuário uma nova tentativa

para escolher outra posição. Após diversos testes de validação da estratégia por meio de simulação aplicou-se o algoritmo no robô real. Observou-se que dinâmica da trajetória obtida pelo robô real corresponde a obtida em simulação, conforme a Figura 28.

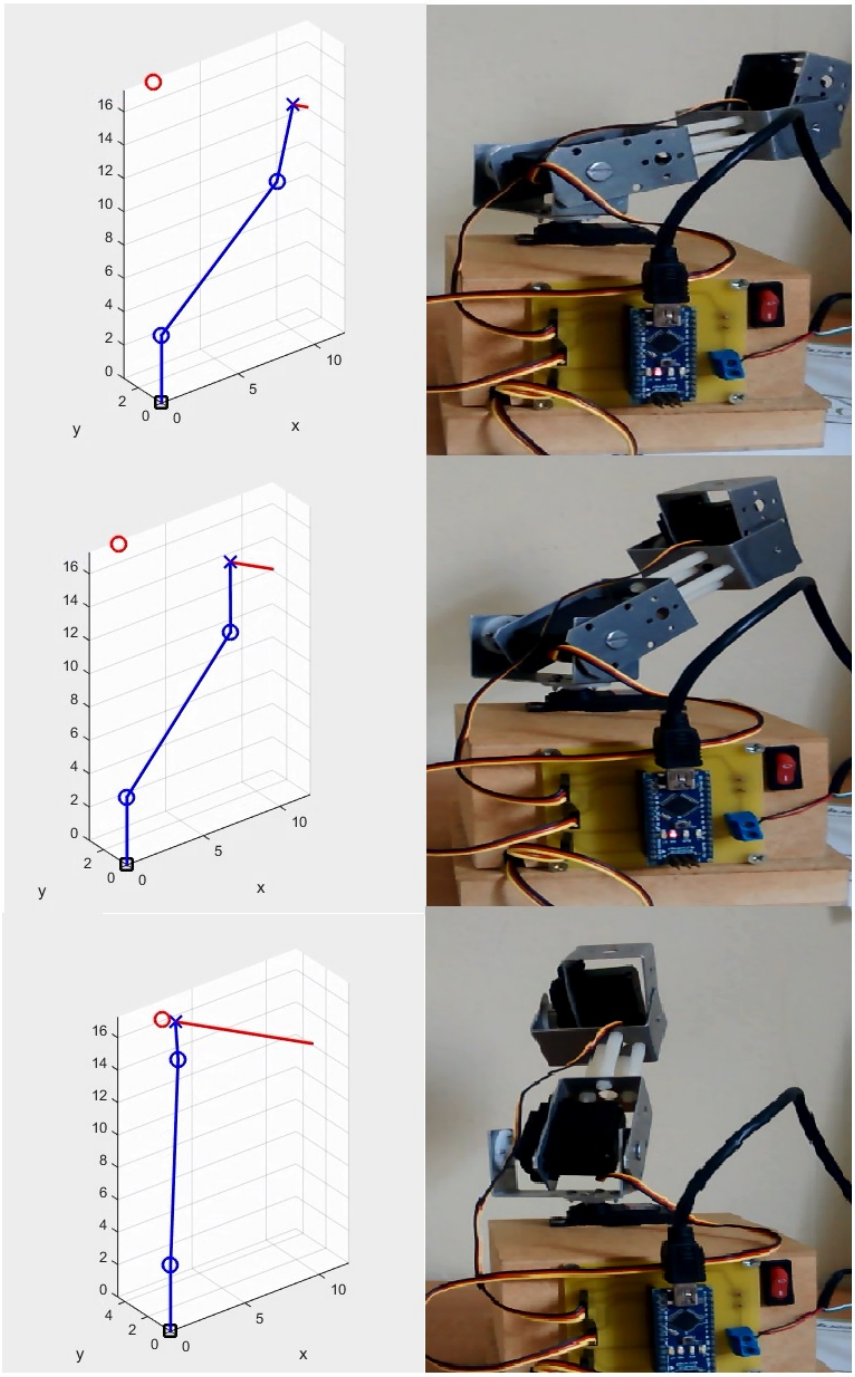

Figura 28 – Validação de experimentos no robô real por campos potenciais.

A Figura 28 mostra três situações diferentes para a validação da estratégia descrita nesta seção. Nota-se que o robô real adquire a configuração mostrada nas indicações de simulação. Este teste foi produzido sem a presença de obstáculos, uma vez que o robô real

não possui sensores capazes de identificar obstáculos. Dessa maneira, a força total F_{tot} é idêntica a força de atração F_{att}.

A validação da estratégia dos pontos próximos aos seus limites é apresentada na Figura 29. Este experimento colocou pontos objetivo fora dos limites, e observou que o caminho do manipulador é permitido até sua situação de limite do espaço de trabalho. A partir desse ponto o algoritmo entra nas restrições e gera as posições aleatórias em torno do ponto objetivo. Caso uma posição válida seja encontrada, esta substitui o ponto objetivo, caso contrário emite-se a mensagem de impossibilidade.

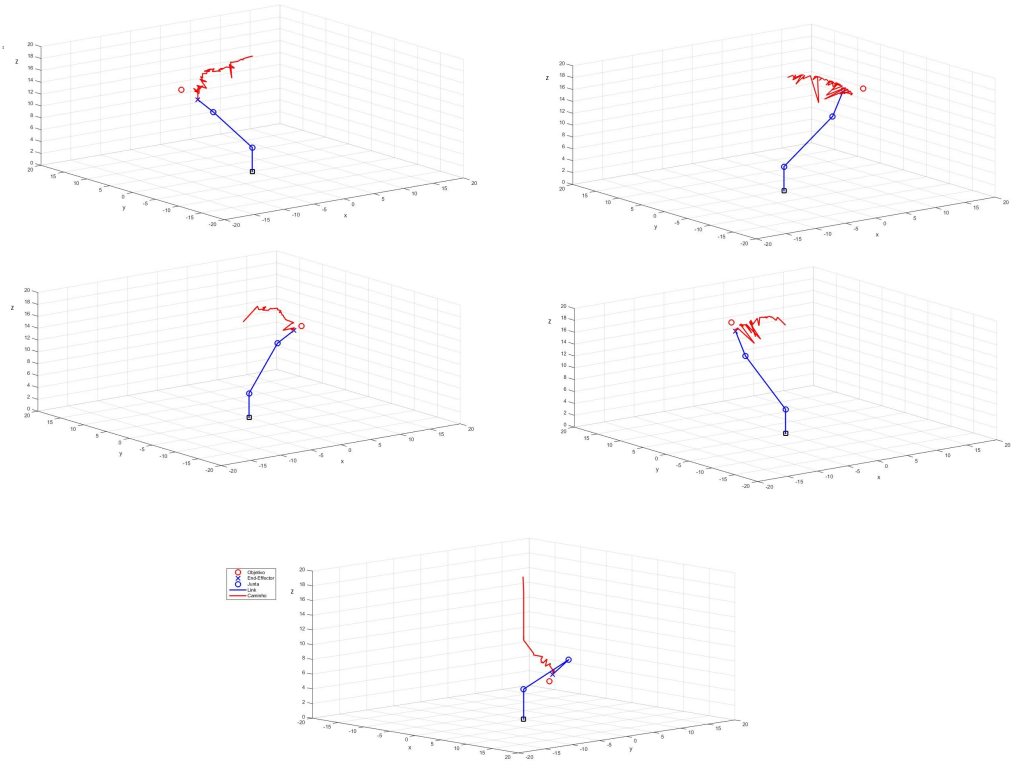

Figura 29 – Desempenho do robô nos limites de seu espaço de trabalho.

4.3 Teleoperação

Uma vez que os algoritmos desenvolvidos tenham sido validados, é possível construir a interface humano-máquina assistiva para usuários com imparidades de movimentos, de modo a possibilitar a teleoperação via Emotiv (Seção 2.2.1, página 17). Conforme descrito em seções anteriores, o Emotiv disponibiliza poucos canais, dos quais escolheram-se quatro: olhar à esquerda, à direita, mordida e levantar as sobrancelhas. O Emotiv se encarrega de reconhecer qual dos movimentos é executado pelo usuário, e envia a classificação por meio de teclas do teclado, as quais podem ser endereçadas conforme a escolha do projetista. No qual é mostrado no Fluxograma da Figura X.

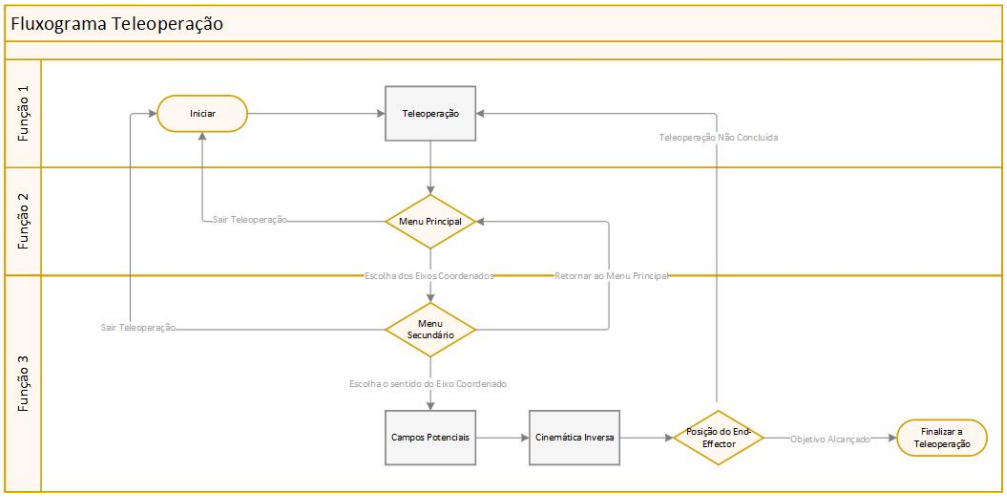

Figura 30 – Fluxograma da Teleoperação.

Dessa maneira, oferece-se ao usuário até quatro comandos diferentes por interface apresentada. É possível ao usuário navegar dentre os diversos menus e submenus criados por meio dos comandos faciais descritos. Neste trabalho endereçou-se o "olhar à esquerda" a tecla "a", "olhar à direita" para a tecla "d", mastigar para a tecla "w" e, finalmente, "levantar as sobrancelhas" para a tecla "s". Sempre que um movimento é reconhecido pelo Emotiv, este emite o sinal do caracter de teclado correspondente para a memória do sistema, bastando ao projetista capturar a tecla e endereçá-la a um comando específico. Os menus criados em Matlab são textuais, e são exibidos na Figura 31.

```
-------------------------------------
Menu Principal:
Escolha dentre as seguintes opções:
A para o eixo x
B para o eixo y
C para o eixo z
D para sair
-------------------------------------
-------------------------------------
Menu Secundário:
Escolha dentre as seguintes opções:
A 1 cm
B -1cm
C Retorna ao menu Principal
D para sair
-------------------------------------
```

Figura 31 – Menu para a Teleoperação.

O primeiro menu, denominado "Menu Principal", oferece quatro opções de movi-

mento do manipulador para o usuário: (A) movimentos no eixo-x, escolhida com “olhar à esquerda”; (B) no eixo-y, escolhida com “olhar à direita”; (C) no eixo-z, escolhida com “mastigar” e, finalmente, (D) sair do programa, escolhida com “levantar as sobrancelhas”.

Uma vez escolhido uma opção de comandos para o robô, um segundo menu é apresentado, desta vez para executar os movimentos do manipulador. Se um usuário escolhe, por exemplo, uma das opções de movimentar o manipulador nos eixos coordenados, o segundo menu, apresentado na Figura 31 é aberto, possibilitando que o usuário possa deslocar o robô no sentido positivo ou negativo do eixo cartesiano escolhido.

O novo menu é comandado da mesma maneira que o primeiro: (A) 1 cm no sentido positivo, escolhida com “olhar à esquerda”; (B) 1 cm no sentido negativo, com “olhar à direita”; (C) retornar ao menu anterior, escolhida com “mastigar” e, finalmente, (D) sair do programa, escolhida com “levantar as sobrancelhas”. Dessa maneira o usuário pode situar o robô na posição que lhe for mais favorável, permitindo a execução de alguma atividade, como pegar um copo, abrir uma porta, e assim por diante.

O algoritmo de campos potenciais foi usado como um controlador de trajetória. Mas a funcionalidade principal desse algoritmo é a capacidade de evitar obstáculos, de forma a considerar a proteção do usuário e do robô. Como o robô real não foi equipado com sensores *rangefinders*, que são capazes de oferecer a distância entre ele e o obstáculo, nele foram implementados somente os campos potenciais atrativo e total, que é idêntico ao atrativo. No entanto, no robô simulado, que possui todas as propriedades cinemáticas do robô real, inclusive as restrições impostas pelos motores, foi possível implementar todos os campos, atrativo, repulsivo e total.

4.3.1 Teleoperação sem obstáculos

A fim de utilizar a estratégia tanto com o robô simulado quanto com o real, tarefas para o usuário foram propostas para analisar não somente o sucesso de integração das técnicas apresentadas, mas também o esforço cognitivo e físico impostos ao usuário utilizando a interface humano-máquina assistiva. Dessa maneira, as tarefas envolveram levar o manipulador de uma determinada posição atual até uma outra final, todas pré-definidas com antecedência.

Primeiramente executou-se a estratégia com o robô simulado. A Figura 32 mostra um exemplo dessas tarefas. Na figura observa-se que a linha vermelha é o caminho percorrido pelo manipulador, e que o “x” preto é o objetivo final a ser alcançado pelo usuário, enquanto o manipulador é dado em azul. Nos exemplos mostrados o usuário teve sucesso em aproximar o robô suficientemente perto do objetivo, com proximidade abaixo de 1 centímetro, sendo assim considerada completa a tarefa. Observam-se pequenas descontinuidades na trajetória, nas quais foram executadas a estratégia de busca aleatória por pontos em torno de uma posição não contemplada pelo espaço de trabalho do robô

com suas restrições. A estratégia de busca de pontos auxiliou o usuário a cumprir a tarefa com sucesso, pois o robô assumiu posições permitidas para possibilitar ao usuário a continuidade da tarefa, sendo este o conceito de controle compartilhado [7, 11].

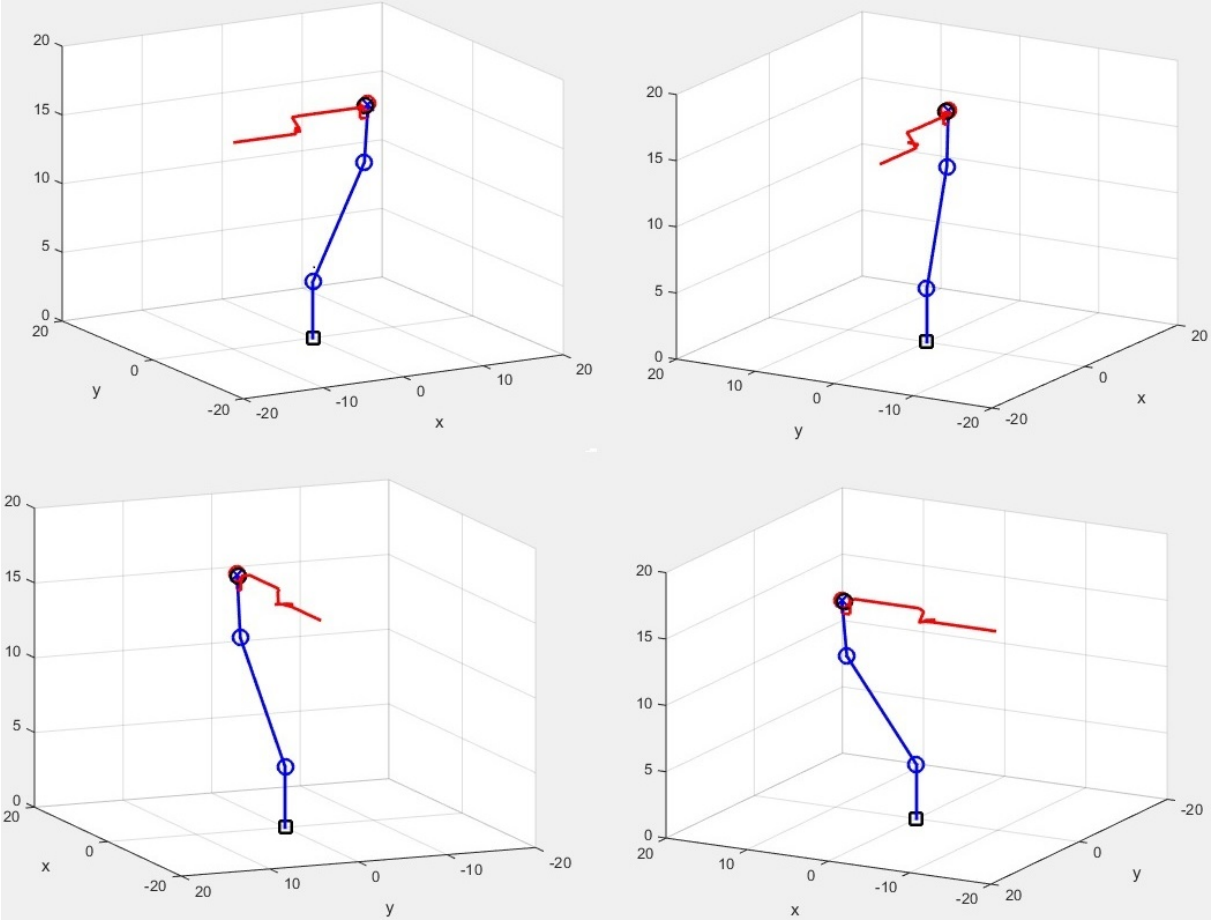

Figura 32 – Simulação da teleoperação do manipulador no cumprimento de tarefas.

O usuário foi capaz de executar a tarefa em 50 segundos, emitindo 40 comandos (média de 0,8 comandos/s). Assim, é possível aplicar a mesma estratégia para o robô real, no qual o usuário sentou-se na parte traseira do manipulador, tendo alvos colocados na parede de maneira a ter visão tipo planta-baixa da situação, conforme a Figura 33. Para realizar os resultados dessa prática foram criados três objetivos, mostrados na mesma figura. O objetivo do usuário é, sem movimentar a cabeça, levar o efetuador do manipulador até os três alvos, em sequência, partindo de uma posição inicial em que o robô possui o braço todo repousado sobre o eixo-z positivo, conforme evidenciado na figura em questão.

A quantidade de comandos e o tempo da tarefa são fundamentais para avaliar a carga de esforço cognitiva e física impostas ao usuário. O treinamento de um usuário é um fator a ser considerado, no entanto, quando se trata de eletromiografia de superfície (sEMG), a adaptação dos usuários é usualmente rápida [11]. A Figura 34 mostra a sequência seguida pelo usuário colocando o efetuador do robô na parte mais central dos alvos, e, conforme é possível observar, as tarefas foram concluídas com êxito. O usuário,

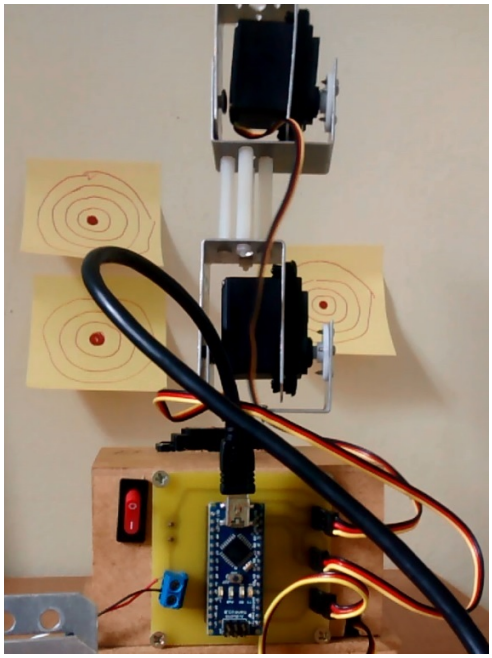

Figura 33 – Tarefas para o usuário com o robô real.

autor deste trabalho, foi capaz de concluir a tarefa com, aproximadamente, 80 comandos em 90 segundos, produzindo uma média de 0,89 movimentos por segundo. Esta média evidencia que o controle teleoperado é possível por meio de interfaces assistivas.

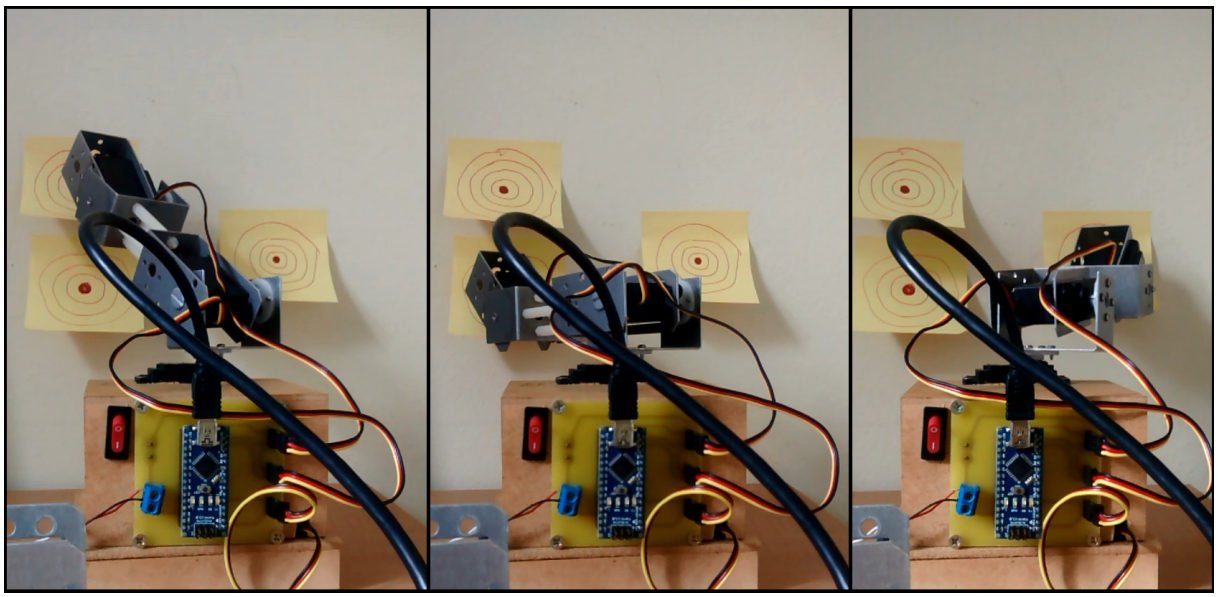

Figura 34 – Teleoperação do robô real.

4.3.2 Robô virtual com obstáculos no ambiente

Em um ambiente virtual, foi modelado um robô capaz de detectar a presença de obstáculos, ou seja, munido de um sensor *rangefinder*. Para tanto, deu-se ao usuário a tarefa de, novamente, levar o robô de um ponto inicial a um objetivo. No entanto, um

obstáculo virtual foi inserido entre os pontos mencionados, de modo a oferecer uma ameaça iminente para o robô. O usuário deve conseguir realizar e teleoperação com a meta de desviar do obstáculo e atingir o objetivo. Essa tarefa é mostrado na Figura 35.

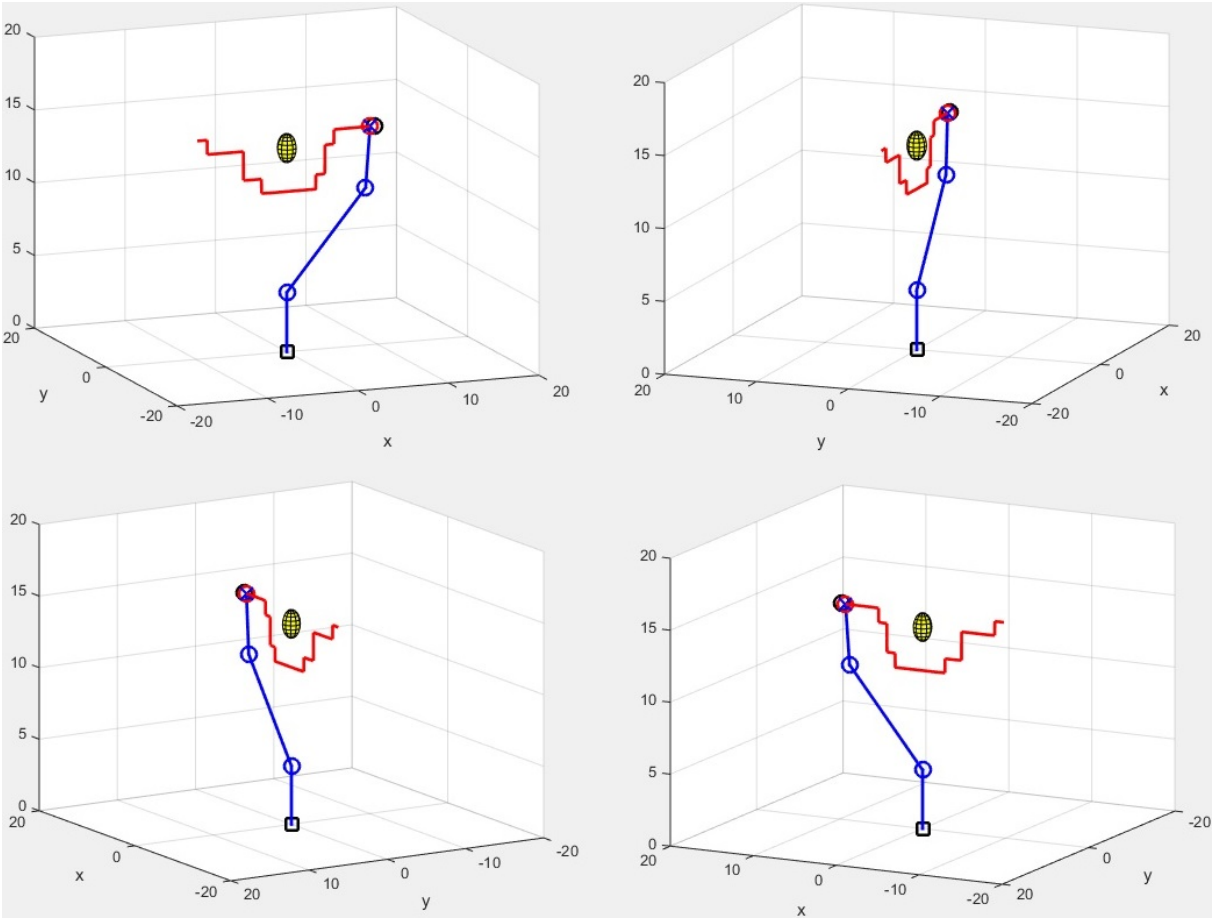

Figura 35 – Teleoperação do robô virtual sem campos potenciais.

Na tarefa proposta os campos potenciais não estavam acionados, ficando a cargo do usuário fazer todas as manobras de desvio de obstáculo. Observa-se que os resultados mostrados na Figura 35 foram satisfatórios, de modo que o usuário conseguiu desviar do obstáculo e atingir o objetivo, mostrando que a interface humano-máquina permitiu o acesso aos comandos. No entanto, como a tarefa requereu muitas trocas de menus, e a quantidade de comandos aumentou consideravelmente, sendo 54 comandos em 60 segundos, com média de 0,9 comandos/s, tornando a tarefa cansativa.

Dessa maneira, propôs-se a utilização dos campos potenciais para verificar se a metodologia de controle compartilhado facilitaria o controle do usuário. Assim, o novo desafio ficou em realizar novamente a tarefa, mas agora com os campos potenciais auxiliando a realizar a teleoperação, assim, o usuário não precisa se preocupar em evitar a colisão, que fica a cargo dos campos potenciais. O objetivo está, portanto a 40 cm afrente do usuário em linha reta e existe um obstáculo no ponto médio do trajeto. A proposição é que o usuário não se importe com esse obstáculo e movimente o robô em linha reta

na direção do objetivo, ignorando sua presença. Os resultados deste experimento são mostrados na Figura 36, na qual é possível observar que a tarefa foi cumprida com sucesso, com o manipulador atingindo o ponto objetivo sem qualquer colisão.

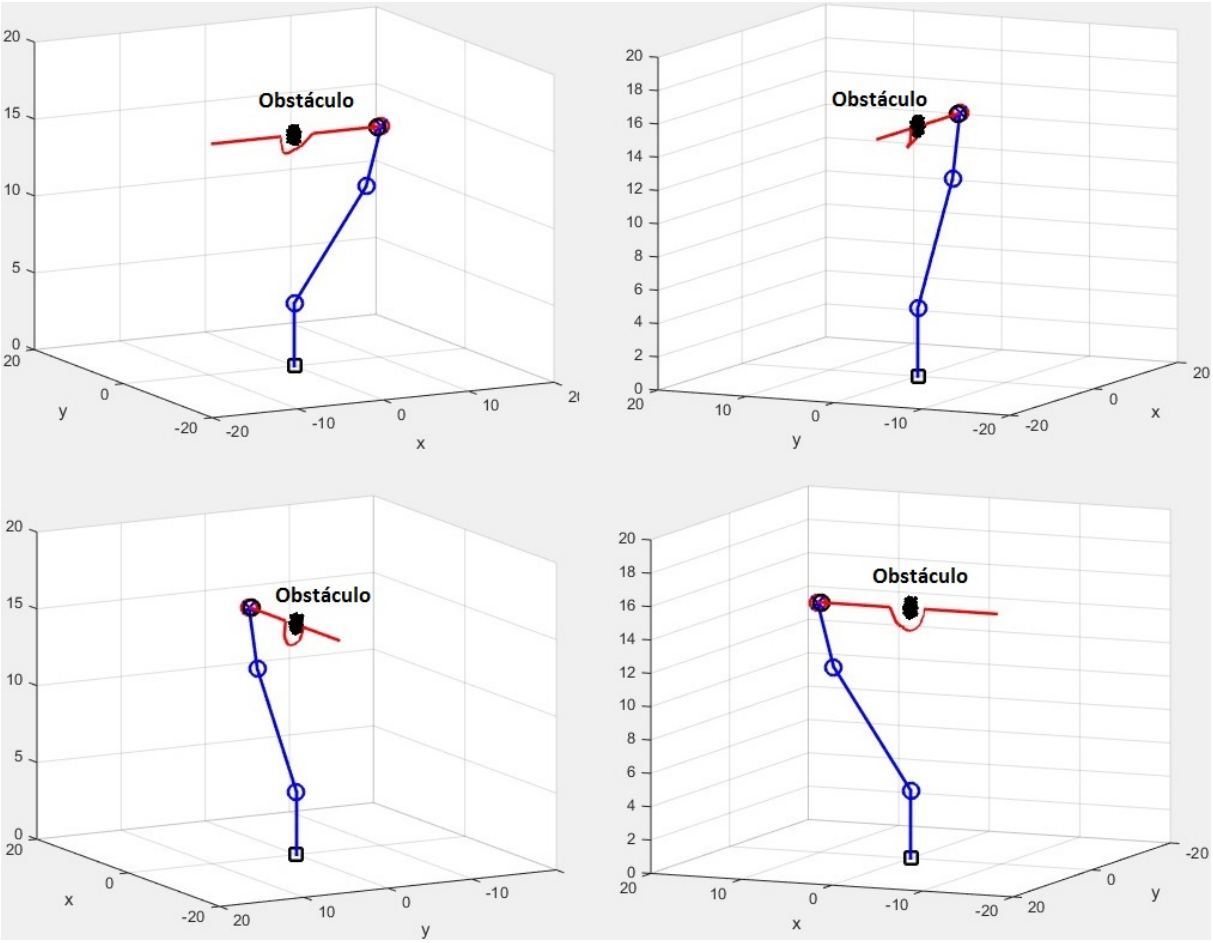

Figura 36 – Teleoperação do virtual com detecção de obstáculos.

Na Figura 36 é possível observar que, por conta do algoritmo dos campos potenciais compartilhando o controle com o usuário, nenhuma colisão ocorreu e, ao comparar os desvios feitos com relação aos realizados pelo próprio usuário (Figura 35), os desvios foram mais incisivos, contornando o obstáculo com maior acurácia e precisão. Ainda mais, o usuário não necessitou realizar nenhum desses desvios, ficando a cargo dos campos potenciais realizar a proteção do robô. Dessa maneira, a carga cognitiva imposta ao usuário caiu drasticamente, com 10 comandos em 30 segundos, média de 0,34 comandos/s, tornando a tarefa menos cansativa, e mais rápida.

O usuário enviou apenas comandos para o manipulado andar um passo positivo no eixo-x. O efetuador está imerso em um campo de percepção que tem um raio de 1 centímetro no qual o mesmo consegue perceber o obstáculo. Dentro desse raio a força repulsiva irá atuar e desviar a ferramenta de ação. Apesar dos bons resultados, este método possui alguns problemas quando usado como controlador autônomo. Se o obstaculo tiver

uma geometria não-convexa (obstáculo em forma de “U”, por exemplo, que criam mínimos locais), o robô ficará preso caso entre o obstáculo pois as forças atrativa e repulsiva se equilibrarão e o robô não poderá sair.

No entanto, a aplicação não é de controle autônomo, e sim, compartilhado. Ou seja, é o usuário quem controla a direção a ser seguida e, portanto, tem condições de tirar o robô de qualquer obstáculo não-convexo, em que os campos potenciais auxiliariam o usuário a evitar a colisão. Dessa maneira, valida-se a estratégia de controle compartilhado desenvolvida, que foi capaz de auxiliar o usuário a não colidir com o ambiente, reduzindo drasticamente a carga cognitiva imposta ao usuário, tornando a tarefa mais simples de ser cumprida. Assim, tem-se uma estratégia em robótica assistiva que efetivamente pode contribuir para usuários recuperarem sua mobilidade.

5 Conclusões

Este trabalho apresentou a construção de um robô manipulador do tipo articulado, e todos os algoritmos necessários para o seu controle e proteção. Além disso, o desenvolvimento de uma estratégia de controle compartilhado para auxiliar um usuário a controlar este manipulador foi desenvolvida, com a intenção de melhorar a mobilidade do usuário. Como resultados, conforme apresentado no Capítulo 4, tanto o robô real quanto o virtual cumpriram os objetivos para os quais foram propostos.

Os resultados mostraram que, dado um ponto objetivo dentro do espaço de trabalho do robô, um usuário utilizando uma interface humano-máquina assistiva tem condições de realizar a tarefa com baixa carga cognitiva. O robô real não foi munido de sensores tipo *rangefinder* e por isso não pode testar seu desempenho em uma trajetória com objetos. No entanto, um robô virtual munido de todas as restrições cinemáticas impostas ao robô real pôde ser utilizado para a verificação das proposições de controle compartilhado, demonstrando êxito no cumprimento das tarefas.

O robô real cumpriu o objetivo da teleoperação, respeitando sua limitação dos ângulos. No entanto, o trabalho não abordou o tema do controle da orientação da ferramenta, apenas da posição. Isto se deu pois, para adicionar ao usuário a tarefa de controlar, além da posição, a orientação da ferramenta, o esforço cognitivo aumentaria bastante, tornando a tarefa cansativa.

No entanto, a estratégia de cálculo da cinemática inversa por Gradiente Descendente possui a capacidade de abordar o controle de posição e orientação ao mesmo tempo, sendo possível oferecer o controle da orientação ao usuário, caso necessário. Ainda assim, é preferível desenvolver alguma estratégia para o controle automático da orientação do efetuador, como, por exemplo, por meio de visão computacional, a adicionar esta carga cognitiva exacerbada ao usuário, tornando a atividade extremamente cansativa.

O objetivo de reduzir a quantidade de comandos que o usuário necessita enviar para deslocar o robô com segurança foi atingida por meio dos campos potenciais, que auxiliam o usuário a evitar colisões, tornando esta estratégia de controle compartilhado uma boa opção para ser implementada em larga escala. Os problemas inerentes do campos potenciais frente a obstáculos não-convexos é resolvida pelo fato do usuário participar do controle e ter a possibilidade de retirar o robô dessas situações em que o algoritmo original dos campos potenciais, quando autônomo, falha.

5.1 Trabalhos Futuros

O trabalho pode ser melhorado implementando por meio de algoritmos de otimização, como o *Twiddle*, programação quadrática ou algoritmos de inteligência computacional,

para ajustar todos os parâmentros do algorítimo de campos potenciais, como o K_{att} e o K_{rep}, buscando obter uma solução ótima no sentido de aumentar a segurança e obter uma trajetória mais suave.

A interface humano-máquina pode ser melhorada, incluindo aspectos gráficos, melhorias na facilidade de escolha e alternância entre menus, tornando-a mais amigável (*user friendly*), com intuito de proporcionar uma teleoperação menos cansativa e mais confortável.

No robô real poderiam ser inseridos sensores exteroceptivos tipo *rangefinders*, como SONARES ou câmeras munidas de *software* de identificação de objetos e extração de distâncias, proporcionando ao robô a percepção do ambiente a sua volta, podendo assim, utilizar a estratégia ao seu máximo, desviando de obstáculos e ajudando o usuário no compartilhamento do controle e realização das tarefas.

REFERÊNCIAS

[1] Emotiv, "Emotiv epoc scientific contextual eeg," https://emotiv.com/, 2016, [Online; acessado em 20-01-2016].

[2] L. R. Olivi, *Notas de aula da disciplina "Manipuladores Robóticos" (ENE124).* Universidade Federal de Juiz de Fora (UFJF), 2016.

[3] ABB, *IRB 1410 Industrial Robot Data Sheet*, 201O, acessado: 15/11/2015.

[4] Arduino, "Arduino nano," https://www.arduino.cc/en/Main/ArduinoBoardNano, 2016, [Online; acessado em 20-02-2016].

[5] DFRobot, "Quality arduino and robotics products," http://www.dfrobot.com/, 2016, [Online; acessado em 20-01-2016].

[6] IBGE, "Instituto brasileiro de geografia e estatística, censo 2010," http://censo2010.ibge.gov.br/, 2010, [Online; acessado em 20-10-2015].

[7] L. Olivi, R. Souza, E. Rohmer, and E. Cardozo, "Shared control for assistive mobile robots based on vector fields," in *IEEE 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, vol. 1, no. 1, 2013, pp. 96–101.

[8] K. Raizer, E. Rohmer, A. Paraense, and R. Gudwin, "Effects of behavior network as a suggestion system to assist bci users," in *IEEE Symposium on Computational Intelligence in Rehabilitation and Assistive Technologies (CIRAT)*, vol. 1, no. 1, 2013, pp. 40–47.

[9] E. Rohmer, P. Pinheiro, K. Raizer, L. Olivi, and E. Cardozo, "A novel platform supporting multiple control strategies for assistive robots," in *24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, vol. 1, no. 1, 2015, pp. 763–769.

[10] J. d R Millán, R. Rupp, G. Müller-Putz, R. Murray-Smith, C. Giugliemma, M. Tangermann, C. Vidaurre, F. Cincotti, A. Kübler, R. Leeb, C. Neuper, K.-R. Müller, and D. Mattia, "Combining brain-computer interfaces and assistive technologies: state-of-the-art and challenges," *Frontiers in neuroscience*, vol. 4, no. 1, pp. 1–15, 2010.

[11] L. Olivi, "Navegação de robôs móveis assistivos por controle compartilhado baseado em campos vetoriais," Ph.D. dissertation, Faculdade de Engenharia Elétrica e Computação (FEEC), Universidade Estadual de Campinas (UNICAMP), 2014.

[12] N. Papanikolopoulos, "Integrating computer vision and control for vision-assisted robotic tasks," *American Control Conference, Proceedings of the 1995*, vol. 1, no. 5, pp. 904 – 908, Jun 1995.

[13] K. Nagai, I. Nakanishi, H. Hanafusa, S. Kawamura, M. Makikawa, and N. Tejima., "Development of an 8 dof robotic orthosis for assisting human upper limb motion," *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 4, no. 6, pp. 3486 – 3491, May 1998.

[14] M. Palankar, K. J. D. Laurentis, R. Alqasemi, E. Veras, and R. Dubey, "Control of a 9-dof wheelchair-mounted robotic arm system using a p300 brain computer interface: Initial experiments," *International Conference on Robotics and Biomimetics*, no. 6, pp. 348–353, Feb 2009.

[15] I. Pathirage, K. Khokarand, E. Klay, R. Alqasemi, and R. Dubey, "A vision based p300 brain computer interface for grasping using a wheelchair-mounted robotic arm," *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, no. 6, pp. 188 – 193, July 2013.

[16] K. George, A. Iniguez, H. Donze, and S. Kizhakkumthala, "Design, implementation and evaluation of a brain-computer interface controlled mechanical arm for rehabilitation," *IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, no. 3, pp. 1326–1328, May 2014.

[17] S. Sanei, *EEG Signal Processing*, 1st ed. Wiley e Sons, 2007.

[18] J. Webb, Z. G. Xiao, K. Aschenbrenner, G. Herrnstadt, and C. Menon, "Towards a portable assistive arm exoskeleton for stroke patient rehabilitation controlled through a brain computer interface," *Biomedical Robotics and Biomechatronics (BioRob)*, no. 6, pp. 1299 – 1304, June 2012.

[19] N. Yamawaki, N. Nakasako, T. Nishimae, T. Shinohara, M. Nakayama, and D. Yoshida., "A prototype of bci-robot arm system with 1ch acoustic distance measurement device," *2013 International Symposium on Intelligent Signal Processing and Communications Systems (ISPACS)*, no. 4, pp. 336 – 339, Nov 2013.

[20] MathWorks, "Matlab - the language of technical computing," http://www.mathworks.com/products/matlab/, 2016, [Online; acessado em 20-01-2016].

[21] Arduino, "Arduino," https://www.arduino.cc/, 2016, [Online; acessado em 20-01-2016].

[22] P. Corke, *Robotics, Vision and Control, Fundamental Algorithms in Matlab*, 1st ed. Springer, 2011.

[23] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics - Modeling, Planing and Control*, 1st ed. Springer, 2009.

[24] J. R. Andrews and N. Hogan, "Impedance control as a framework for implementing obstacle avoidance in a manipulator," in *Control of Manufacturing Processes and Robotic Systems*, vol. 1, no. 1, 1983, pp. 243–251.

[25] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *IEEE International Conference on Robotics and Automation*, vol. 1, no. 1, 1985, pp. 500–505.

ANEXO A – Posição das Juntas

```
function [ f_J1 ] = firtjoin( l1 )
f_J1 =[0;0;l1];

end


function [ f_J2] = secondjoin(l1,l2,th1,th2 )
f_J2=[l2*cos(th1)*cos(th2);
l2*cos(th2)*sin(th1);
l1 + l2*sin(th2)];

end


function [ f_J3 ] = thirdjoin( l1,l2,l3,th1,th2,th3 )
f_J3=[cos(th1)*(l3*cos(th2 + th3) + l2*cos(th2));
sin(th1)*(l3*cos(th2 + th3) + l2*cos(th2));
l1 + l3*sin(th2 + th3) + l2*sin(th2)];


end
```

ANEXO B – Jacobiano

```
function [ Jf ] = jacobiano( l1,l2,l3,th1,th2,th3 )
Jf=[−sin(th1)*(l3*cos(th2 + th3) + l2*cos(th2)),
 −cos(th1)*(l3*sin(th2 + th3) + l2*sin(th2)), −l3*sin(th2 + th3)*cos(th1);
cos(th1)*(l3*cos(th2 + th3) + l2*cos(th2)), −sin(th1)*(l3*sin(th2 + th3) +
 l2*sin(th2)), −l3*sin(th2 + th3)*sin(th1);
0,             l3*cos(th2 + th3) + l2*cos(th2),           l3*cos(th2 + th3)];
end
```

ANEXO C – Cinemática Inversa

```
function [ Theta ,k] = cinematicainvdegtra( G ,T )

Theta = T;
%% Tamanho dos links do braco
L1 = 4.07;
L2 = 11.05;
L3 = 4.21;

%% Parametros

ksi = 1e-4;
n=0.2;
k=0;

%% Posicao da Ferramenta de Trabalho
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2),Theta(3)));
EQM = mean((G - P_J3).^2);
while EQM > ksi
GradEQM = -(G - P_J3);
J = double(jacobiano(L1,L2,L3,Theta(1),Theta(2), Theta(3)));
if det(J) == 0
J = J + 1e-5 * eye(size(J));
end
GradTh = inv(J) * GradEQM;
Theta = Theta - n * GradTh;
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2),Theta(3)));
%% Erro Quadratico Medio atual
EQM = mean((G - P_J3).^2);
k=k+1;
t =toc;
if(t>0.1)
EQM = ksi;
end

end

end
```

ANEXO D – Campos Potenciais Robô Real

```
function [ g, Theta, Thetadeg ] = CamposPotenciaisRobo( Theta, G)

%% Tamanho dos links do braco
L1 = 4.07;
L2 = 11.05;
L3 = 4.21;

%% Constante do campo atrativo
Katt = 0.01;

%%
deltad = 0.1;
ksi1 = 0.07;
Cont = 0;

%% Gaus
n = 50;
X = 1:n;
delt_s =3;
gx = gaussmf(X,[delt_s 0]);
gy =gaussmf(X,[delt_s 0]);
gz =gaussmf(X,[delt_s 0]);
deltx = gx*(rand(n)-rand(n));
delty = gy*(rand(n)-rand(n));
deltz = gz*(rand(n)-rand(n));

%% Posicao das Juntas
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));

%% Vetor Aleatorio
V=[];

%% Campos Potenciais

Dpg = ((P_J3-G)'*(P_J3-G)).^(1/2);

while Dpg > ksi1
Fatt = -Katt.*(P_J3 - G);
%% Sem Forca Repulsiva


%%
Ftot = Fatt;
g = P_J3 + (deltad/norm(Ftot)) .* Ftot;
```

```
Dpg = ((P_J3-G)'*(P_J3-G)).^(1/2);

if Dpg < deltad
g = G;
end
Theta = cinematicainvdegtra(g,Theta);
Theta = mod(Theta , 2*pi);

if (max(Theta(1) > pi) < 1)&&(max(Theta(2) > pi) < 1)
&& (max(Theta(3) > 1.7453) < 1)
Thetadeg = round(rad2deg(Theta));
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));

else
Cont = Cont+1;
for i=1:n
g1 = g + [deltx(i);delty(i);deltz(i)];
Theta = cinematicainvdegtra(g1,Theta);
Theta = mod(Theta , 2*pi);
if (max(Theta(1) > pi) < 1)&&(max(Theta(2) > pi) < 1)
&& (max(Theta(3) > 1.7453) < 1)
V=[V,Theta];
end
end
C = size(V);
ind =C(2)-1;
if isempty(V) == 0
Theta = V(:,round(ind)+1);
Thetadeg = round(rad2deg(Theta));
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));
V=[];
if(Cont>2)
Dpg = ksi1;
end
else
Dpg = ksi1;
end

end
end
```

ANEXO E – Teleoperação com o Robô Real

```
%%
a.servoAttach(3);
a.servoAttach(10);
a.servoAttach(11);



%% Posicao Inicial
Theta = [pi/2;pi/2;0];
Thetadeg = round(rad2deg(Theta));
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));

%%
fprintf('\nAperte A para comecar ');
%% Posicao da base do robo
base.x = 0;
base.y = 0;
base.z = 0;

%% Tamanho dos links do braco
L1 = 4.07;
L2 = 11.05;
L3 = 4.21;

%% Objetivo Inicial
G=[0;0;19.33];

%% Objetivo Global

Gb = [5;8;15];

%% Passo do Manipulador
d = 1;%cm
%% Posicao Inicial Das Juntas Do Robo
P_J1 = double(firtjoin(L1));
P_J2 = double(secondjoin(L1,L2,Theta(1),Theta(2)));
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));
%% Vetor que Guarda Posicao
Pose=[];
%%
st = getkey;
if (st == 97)
```

```
fprintf('\nStart');
end


while (st == 97)
sprintf('\nMenu Principal:')
ei= getkey;
switch ei
case 97
sprintf('\nMenu Secundario:')
se=getkey;
switch se
case 97
fprintf('FRONT')
G = G + [ d ; 0 ; 0];
[ g, Theta ,  Thetadeg ] = CamposPotenciaisRobo( Theta, G);
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));
G = g;

case 98
fprintf('BACK')
G = G + [-d ; 0 ; 0];
[ g, Theta ,  Thetadeg ] = CamposPotenciaisRobo( Theta, G);
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));
G = g;

case 99
st = 97;
case 100
st = 0;
end
case 98
sprintf('\nMenu Secundario:')
se=getkey;
switch se
case 97
fprintf('RIGHT')
G = G + [0;d; 0];
[ g, Theta ,  Thetadeg ] = CamposPotenciaisRobo( Theta, G);
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));
G = g;
```

```
case 98
fprintf('LEFT')
G = G + [0 ;-d ;0 ];
[ g, Theta ,  Thetadeg ] = CamposPotenciaisRobo( Theta, G);
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));
G = g;

case 99
st = 97;
case 100
st = 0;
end
case 99
sprintf('\nMenu Secundario:')
se=getkey;
switch se
case 97
fprintf('UP')
G = G + [0 ; 0; d];
[ g, Theta ,  Thetadeg ] = CamposPotenciaisRobo( Theta, G);
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));
G = g;

case 98
fprintf('DOWN')
G = G + [0; 0; -d];
[ g, Theta ,  Thetadeg ] = CamposPotenciaisRobo( Theta, G);
a.servoWrite(3,Thetadeg(1));
a.servoWrite(10,Thetadeg(2));
a.servoWrite(11,Thetadeg(3));
G = g;

case 99
st = 97;
case 100
st = 0;
end
case 100
st = 0;
end
if(abs(G-Gb))<0.5
disp('Objetivo Alcancado')
```

```
st = 0;

end

end

disp('FIM')
```

ANEXO F – Campos Potenciais Robô Virtual

```
function [ g, Theta , Pose] = CamposPotenciaisRobo( Theta, G,Pose,Gb,obst )

%% Posicao da base do robo
base.x = 0;
base.y = 0;
base.z = 0;
%% Tamanho dos links do braco
L1 = 4.07;
L2 = 11.05;
L3 = 4.21;

%% Constante do campo atrativo
Katt = 0.01;

%% Constante de campo repulsivo
Krep =  7e8;

%% Horizonte de eventos
eps0 = 1.1;

%% Parametros
deltad = 0.1;
ksi1 = 0.07;
Cont = 0;

%% Gaus

n = 50;
X = 1:n;
delt_s =3;
gx = gaussmf(X,[delt_s 0]);
gy =gaussmf(X,[delt_s 0]);
gz =gaussmf(X,[delt_s 0]);
deltx = gx*(rand(n)-rand(n));
delty = gy*(rand(n)-rand(n));
deltz = gz*(rand(n)-rand(n));

%% Posicao das Juntas
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));

%% Vetor Aleatorio
V=[];

%% Vetor para Plot
```

```
Obsta = [];

%% Inicialisacao das forcas em zero
Frep = [0;0;0];
Fatt =[0;0;0];
epsi=[];

%% Campos Potenciais
Dpg = ((P_J3-G)'*(P_J3-G)).^(1/2);

while Dpg > ksi1
Fatt = -Katt.*(P_J3 - G);
Frep = [0;0;0];

for q = 1:100
epsi(:,q)=((P_J3-obst(:,q))'*(P_J3-obst(:,q))).^(1/2);
end

indices = find(epsi <= eps0);
if isempty(indices) == 0
for j = 1:size(indices,2)
a = (1/epsi(indices(j)))^3;
b = (1/epsi(indices(j))) - (1/eps0);
Frepj = (a*b).*(P_J3 - obst(:,indices(j)));
Frep = Frep + Frepj;

end
Frep = Krep * Frep;

end
Obsta = [Obsta,[obst(1,indices);obst(2,indices)]];
Ftot = Fatt + Frep;%Forca total
g = P_J3 + (deltad/norm(Ftot)) .* Ftot;
Dpg = ((P_J3-G)'*(P_J3-G)).^(1/2);
if Dpg < deltad
g = G;
end
Theta = cinematicainvdegtra(g,Theta);
if (max(Theta > 2*pi) < 1)
P_J1 = double(firtjoin(L1));
P_J2 = double(secondjoin(L1,L2,Theta(1),Theta(2)));
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));
Pose = [Pose , P_J3];
%% Simulacao
plot3(G(1),G(2),G(3),'or','linewidth',2,'markersize',10)
hold on
plot3(Gb(1),Gb(2),Gb(3),'ok','linewidth',2,'markersize',10)
```

```
plot3(base.x , base.y, base.z , 'sk' , 'linewidth' , 2 , 'markersize' , 10)
plot3(P_J1(1) , P_J1(2) , P_J1(3) , 'ob' ,
 'linewidth' , 2 , 'markersize' , 10)
plot3([base.x P_J1(1)] , [base.y P_J1(2)],
[base.z P_J1(3)] , 'b' , 'linewidth' , 2 , 'markersize' , 10)
plot3(P_J2(1) , P_J2(2) , P_J2(3) , 'ob' ,
 'linewidth' , 2 , 'markersize' , 10)
plot3([P_J1(1) P_J2(1)] , [P_J1(2) P_J2(2)] , [P_J1(3) P_J2(3)] , 'b' ,
'linewidth' , 2 , 'markersize' , 10)
plot3(P_J3(1) , P_J3(2),P_J3(3) , 'xb' , 'linewidth' ,
 2 , 'markersize' , 10)
plot3([P_J2(1) P_J3(1)] , [P_J2(2) P_J3(2)], [P_J2(3) P_J3(3)], 'b' ,
 'linewidth' , 2 , 'markersize' , 10)
plot3(Pose(1,:),Pose(2,:),Pose(3,:),'r' , 'linewidth' , 2 , 'markersize' , 10)
plot3(obst(1,:),obst(2,:),obst(3,:),'.k')
hold off
grid
xlabel('x')
ylabel('y')
axis([-20 20 -20 20 0 20 -1 0])
drawnow

else
Cont = Cont+1;

for i=1:n
g1 = g + [deltx(i);delty(i);deltz(i)];
Theta = cinematicainvdegtra(g1,Theta);
Theta = mod(Theta , 2*pi);
if (max(Theta(1) > pi) < 1)&&(max(Theta(2) > pi) < 1)
&& (max(Theta(3) > 1.7453) < 1)% Restricoes do Servo Motor
V=[V,Theta];
end
end
C = size(V);
ind =C(2)-1;
if isempty(V) == 0
disp('=D')
Theta = V(:,round(ind)+1);
P_J1 = double(firtjoin(L1));
P_J2 = double(secondjoin(L1,L2,Theta(1),Theta(2)));
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));
Pose = [Pose , P_J3];
%% Simulacao
plot3(G(1),G(2),G(3),'or','linewidth',2,'markersize',10)
hold on
plot3(Gb(1),Gb(2),Gb(3),'ok','linewidth',2,'markersize',10)
```

```
plot3(base.x , base.y, base.z , 'sk' , 'linewidth' , 2 , 'markersize' , 10)
plot3(P_J1(1) , P_J1(2) , P_J1(3) , 'ob' ,
 'linewidth' , 2 , 'markersize' , 10)
plot3([base.x P_J1(1)] , [base.y P_J1(2)],[base.z P_J1(3)] ,
 'b' , 'linewidth' , 2 , 'markersize' , 10)
plot3(P_J2(1) , P_J2(2) , P_J2(3) , 'ob' ,
'linewidth' , 2 , 'markersize' , 10)
plot3([P_J1(1) P_J2(1)] , [P_J1(2) P_J2(2)] , [P_J1(3) P_J2(3)] , 'b' ,
'linewidth' , 2 , 'markersize' , 10)
plot3(P_J3(1) , P_J3(2),P_J3(3) , 'xb' ,
'linewidth' , 2 , 'markersize' , 10)
plot3([P_J2(1) P_J3(1)] , [P_J2(2) P_J3(2)], [P_J2(3) P_J3(3)], 'b' ,
'linewidth' , 2 , 'markersize' , 10)
plot3(Pose(1,:),Pose(2,:),Pose(3,:),'r' , 'linewidth' , 2 , 'markersize' , 10)
plot3(obst(1,:),obst(2,:),obst(3,:),'.k')
hold off
grid
xlabel('x')
ylabel('y')
axis([-20 20 -20 20 0 20 -1 0])
drawnow
%%
V=[];
if(Cont>2)
disp('Posicao inalcancavel, escolha outra')
Dpg = ksi1;
end
else
disp('Posicao inalcancavel, escolha outra')
Dpg = ksi1;

end

end

end
```

ANEXO G – Teleoperação Robô Virtual

```
clc,close all,clear all

%% Posicao Inicial
Theta = [3.1312; 0.6689; 0.6396];

%%
fprintf('\nAperte A para comecar ');
%% Posicao da base do robo
base.x = 0;
base.y = 0;
base.z = 0;

%% Tamanho dos links do braco
L1 = 4.07;
L2 = 11.05;
L3 = 4.21;

%% Objetivo Inicial
G = [-9.7458; 0.0975; 14.9947];

%% Objetivo Global
Gb = [9.7458; 0.0975; 14.9947];

%% Obstaculo
nob = 1000;
Xob = 1:nob;
delt_sobb =1;
gx = gaussmf(Xob,[delt_sobb 0]);
gy =gaussmf(Xob,[delt_sobb 0]);
gz =gaussmf(Xob,[delt_sobb 0]);
obstx = gx*(rand(nob)-rand(nob));
obsty = gy*(rand(nob)-rand(nob));
obstz = gz*(rand(nob)-rand(nob));
obst = [obstx;obsty;obstz];
for p=1:nob
obst(:,p) = obst(:,p) + [0;0;15];
end

%% Passo do Manipulador
d = 4;
%% Posicao Inicial Das Juntas Do Robo
P_J1 = double(firtjoin(L1));
P_J2 = double(secondjoin(L1,L2,Theta(1),Theta(2)));
P_J3 = double(thirdjoin(L1,L2,L3,Theta(1),Theta(2), Theta(3)));
```

```
%% Vetor que Guarda Posicao
Pose=[];
%%
st = getkey;
if (st == 97)
fprintf('\nStart');
end

%% Simulacao
plot3(G(1),G(2),G(3),'or','linewidth',2,'markersize',10)
hold on
plot3(base.x , base.y, base.z , 'sk' , 'linewidth' , 2 , 'markersize' , 10)
plot3(P_J1(1) , P_J1(2) , P_J1(3) , 'ob' , 'linewidth' , 2 , 'markersize' , 10)
plot3([base.x P_J1(1)] , [base.y P_J1(2)],[base.z P_J1(3)] ,
 'b' , 'linewidth' , 2 , 'markersize' , 10)
plot3(P_J2(1) , P_J2(2) , P_J2(3) ,
 'ob' , 'linewidth' , 2 , 'markersize' , 10)
plot3([P_J1(1) P_J2(1)] , [P_J1(2) P_J2(2)] , [P_J1(3) P_J2(3)] , 'b' ,
'linewidth' , 2 , 'markersize' , 10)
plot3(P_J3(1) , P_J3(2),P_J3(3) ,
 'xb' , 'linewidth' , 2 , 'markersize' , 10)
plot3([P_J2(1) P_J3(1)] , [P_J2(2) P_J3(2)], [P_J2(3) P_J3(3)], 'b' ,
 'linewidth' , 2 , 'markersize' , 10)
plot3(Gb(1),Gb(2),Gb(3),'ok', 'linewidth' ,
 2 , 'markersize' , 10)
plot3(obst(1,:),obst(2,:),obst(3,:),'.k')
hold off
grid
xlabel('x')
ylabel('y')
axis([-20 20 -20 20 0 20 -1 0])
drawnow

while (st == 97)
sprintf('\nMenu Principal:')
ei= getkey;
switch ei
case 97
sprintf('Menu Secundario:')
se=getkey;
switch se
case 97
fprintf('FRONT')
G = G + [ d ; 0 ; 0];
[ g, Theta ,Pose] = CamposPotenciaisRobo( Theta, G,Pose ,Gb,obst);
G = g;
```

```
case 98
fprintf('BACK')
G = G + [-d ; 0 ; 0];
[ g, Theta ,Pose] = CamposPotenciaisRobo( Theta, G,Pose ,Gb,obst);
G = g;

case 99
st = 97;
case 100
st = 0;
end
case 98
sprintf('\nMenu Secundario:')
se=getkey;
switch se
case 97
fprintf('RIGHT')
G = G + [0;d; 0];
[ g, Theta ,Pose] = CamposPotenciaisRobo( Theta, G,Pose ,Gb,obst);
G = g;

case 98
fprintf('LEFT')
G = G + [0 ;-d ;0 ];
[ g, Theta ,Pose] = CamposPotenciaisRobo( Theta, G,Pose ,Gb,obst);
G = g;

case 99
st = 97;
case 100
st = 0;
end
case 99
sprintf('\nMenu Secundario:')
se=getkey;
switch se
case 97
fprintf('UP')
G = G + [0 ; 0; d];
[ g, Theta ,Pose] = CamposPotenciaisRobo( Theta, G,Pose ,Gb,obst);
G = g;

case 98
fprintf('DOWN')
G = G + [0; 0; -d];
[ g, Theta  ,Pose] = CamposPotenciaisRobo( Theta, G,Pose ,Gb,obst);
G = g;
```

```
case 99
st = 97;

case 100
st = 0;
end
case 100
st = 0;
end
if(abs(G—Gb))<0.5
disp('Objetivo Alcancado')
st = 0;

end

end

disp('FIM')
```