UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
CURSO DE ENGENHARIA MECATRÔNICA

RODRIGO ISMAEL DA CONCEIÇÃO

DESENVOLVIMENTO DE UM PROTÓTIPO DE MANIPULADOR ROBÓTICO SUBAQUÁTICO ACOPLÁVEL A UM ROV

Joinville
2018

RODRIGO ISMAEL DA CONCEIÇÃO

DESENVOLVIMENTO DE UM PROTÓTIPO DE MANIPULADOR ROBÓTICO SUBAQUÁTICO ACOPLÁVEL A UM ROV

Trabalho de Conclusão de Curso apresentado como requisito parcial para obtenção do título de Bacharel em Engenharia Mecatrônica, no curso Engenharia Mecatrônica da Universidade Federal de Santa Catarina, Centro Tecnológico de Joinville.

Orientador: Prof. Dr. Lucas Weihmann

Joinville
2018

RESUMO

O trabalho teve como objetivo, desenvolver um protótipo de manipulador robótico que pudesse ser acoplado a um ROV. O manipulador foi projetado levando em consideração uma missão fictícia de ROV, a qual consiste em manipulador objetos no fundo de uma piscina. Para isso, foi idealizado um manipulador de 5 funções tele-operado, o qual utiliza um *joystick* de forma a controlar o manipulador junta a junta. Com base nas experiências e dificuldades enfrentadas durante o projeto, produziu-se um material em que tais percepções pudessem ser consultadas por terceiros para a construção de manipuladores robóticos não-experimentais. O manipulador foi projetado, construído e testado em bancada, utilizando *softwares* e bibliotecas *open-source*, com o menor uso possível de tecnologia proprietária. Ao final do trabalho foram apresentados os resultados obtidos, deixando claro que não houveram testes no ambiente subaquático e que esta deveria ser a continuação lógica, caso hajam trabalhos futuros. Também foi sugerida uma melhoria no método de controle do manipulador o qual trocasse o controle junta a junta, pela cinemática inversa do manipulador.

Palavras-chave: ROV. Manipulador. Robótica.

ABSTRACT

The objective of the work was to develop a prototype of robotic manipulator that can be coupled to an ROV. The manipulator was built-in around a fictional ROV mission that consists of manipulating objects at the bottom of a pool. For this, a tele operated 5-function manipulator was devised, that uses a joystick to control the manipulator joint-by-joint. Based on the experiences and difficulties encountered during the project, a material was produced in which such perceptions can be consulted by third parties for the construction of non-experimental robotic manipulators. The manipulator was projected, built and tested using softwares and open-source libraries, with the least possible use of proprietary technology. In the end of this work are presented the results obtained, making it clear that don't happened subaquatic tests with the manipulator and it's the logical next step in case of further works. It has also been suggested an improvement in the control method of the manipulator, that is, change the joint by joint control by the inverse kinematics of the manipulator.

Keywords: ROV. Manipulator. Robotics

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

DIN Deutsches Institut für Normung

GdL Graus de Liberdade

IEC International Electrotechnical Commission

NEMA National Electrical Manufacturers Association

OCROV Observation Class Remotely Operated Vehicle

PWM Pulse Width Modulation

ROV Remotely Operated Vehicle

RPI Raspberry Pi

VNC Virtual Network Computing

SUMÁRIO

1 INTRODUÇÃO

Mares e oceanos cobrem cerca de 2/3 do globo terrestre e fornecem uma gama de recursos como alimento e petróleo (ANTONELLI; KHATIB, 2008), contudo mesmo este sendo um ambiente tão rico e de tamanha importância para a humanidade, se conhece muito pouco sobre o mesmo, segundo o NOAA (2018) os oceanos são a alma do planeta Terra, sendo responsáveis por regular o clima e prover suporte a todos os organismos vivos, porém, mais de 80 porcento dele permanece não mapeando, não observado e inexplorado, o mesmo órgão afirma ainda que apenas 35 porcento das águas costeiras dos Estados Unidos da América foram mapeadas com dispositivos de alta tecnologia, o que pode parecer estranho visto que este país é uma das potencias econômicas e tecnológicas mundiais.

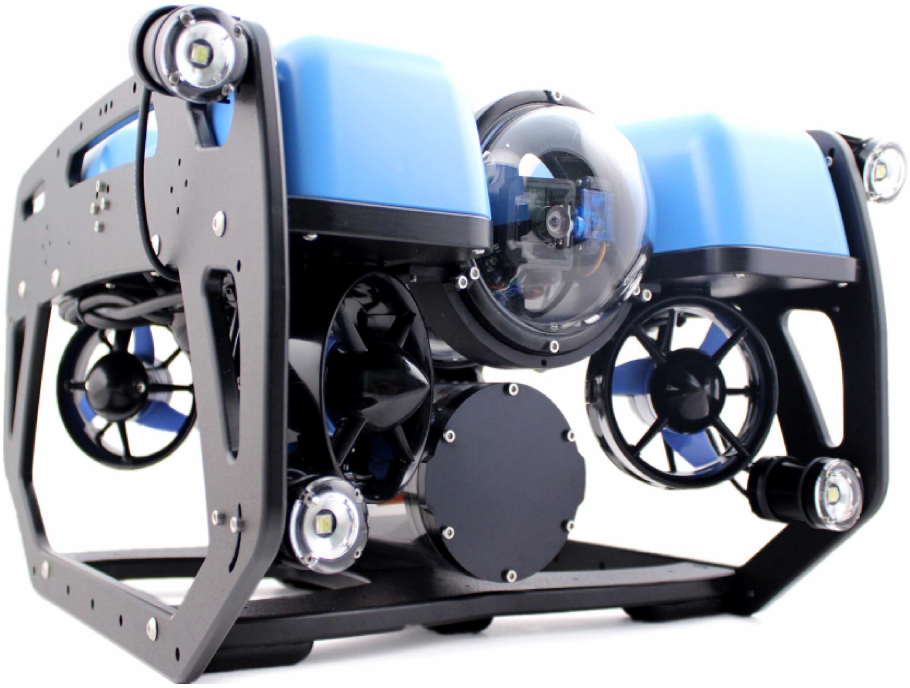
A exploração subaquática pode ser considerada tão perigosa quanto a exploração espacial, uma vez que ambas possuem um ambiente nocivo ao ser humano o qual é desprovido de oxigênio. A exploração subaquática tem ainda outros agravantes como a temperatura reduzida, visão nula ou reduzida, animais marinhos selvagens, ausência de comunicação *wireless* e pressão crescente conforme a profundidade; todos estes fatores tornam o envio de seres humanos um risco muito grande a sua integridade física. Tais adversidades faziam com que a indústria naval e *offshore* tivesse seu crescimento freado e assim surgiu a necessidade da criação de robôs que pudessem exercer as mesmas atividades que mergulhadores sem oferecer risco a suas vidas. Desta forma por volta dos anos 1950's a robótica subaquática foi impulsionada sendo criadas diversas tecnologias, sendo um dos ramos os Veículos Operados Remotamente (ROV's).

Um ROV é um tipo de robô subaquático que possui como principal característica a comunicação entre operador e robô realizada via cabo umbilical, ROVs são utilizados principalmente em operações na indústria naval e *offshore*, como por exemplo, em intervenções subaquáticas e na solda de cascos de navios (ANTONELLI, 2006). Segundo Christ e Wernli (2013), para realizar tais operações os ROV's possuem ao menos um manipulador robótico de cadeia serial, constituído por uma cadeia de elos ligados a juntas rotativas ou prismáticas, bem como pela ferramenta acoplada ao último elo do braço, sendo esta em sua maioria uma ventosa ou uma pinça.

Em meados de 2016 a Universidade Federal de Santa Catarina (UFSC) Campus Joinville, adquiriu um OCROV da empresa Blue Robotics modelo BlueROV2, como o mostrado na figura 1, com intuito de promover a pesquisa e desenvolvimento da

área de robótica subaquática nos cursos de Bacharelado em Engenharia Mecatrônica e Bacharelado em Engenharia Naval. Neste mesmo ano e nos posteriores, as principais atividades que o BlueROV2 executou foram as de inspeção de cascos de navio em portos e a inspeção de tubulações marítimas da PETROBRAS. O site do fabricante do ROV elenca algumas de suas características (ROBOTICS, 2018):

Figura 1 – ROV Modelo BlueROV2 do fabricante Blue Robotics.



Fonte: (ROBOTICS, 2018).

- Vídeo com resolução 1080p HD;
- Disposição de propulsores vetorizados para alta manobrabilidade;
- Estável e otimizado para missões das classes inspeção e pesquisa;
- Fácil de usar, possui plataforma integrada com interface do usuário;
- Altamente expansível com 3 penetradores de cabos livres e pontos de montagem adicionais;
- Cabo umbilical padrão para até 100m de profundidade;
- 6 propulsores e ESCs de alta relação propulsão-peso;
- 2 pares de iluminação LED de 1500 lumens cada, brilho semelhante ao de um farol de automóvel;
- Bateria de fácil substituição;
- Código fonte livre;
- Controlado via *Joystick*.

Contudo, este ROV de pequeno porte não possui nenhum manipulador robótico sendo utilizado apenas para observação, o que o impede de realizar qualquer tipo de intervenção subaquática, limitando sua utilização. No intuito de garantir maior

funcionalidade ao BlueROV2, será proposto um manipulador robótico que possa ser acoplado ao mesmo, o qual deve ser capaz de realizar operações de coleta de pequenos artefatos no ambiente subaquático. O manipulador será projetado de forma que possa ser acoplado ao módulo auxiliar do BlueROV2, o qual é montado abaixo do mesmo, tal módulo possui um casulo extra onde toda a eletrônica e bateria podem ser armazenados em segurança, sem contato com a água. O manipulador proposto deve trocar informações com a superfície via o cabo umbilical utilizado pelo próprio ROV e também deve ser controlado da a partir da superfície via *Joystick* junta a junta.

Inicialmente serão levantados os requisitos do projeto, sendo estes baseados na missão a que o ROV deve cumprir e também nos componentes e dispositivos que podem operar no ambiente subaquático ou técnicas de como torná-los operáveis neste ambiente. Em posse dos requisitos do projeto, a cadeia cinemática é definida, determinando a quantidade e tipos de elos e juntas presentes no manipulador, bem como a ferramenta presente no efetuador final.

Por fim, um código em linguagem Python será desenvolvido para realizar o controle do manipulador e então realizada a integração de software e hardware, seguida por diversos testes, a fim de validar os requisitos de projeto estabelecidos, bem como o funcionamento do sistema.

1.1 Objetivo Geral

Este trabalho propõe o desenvolvimento de um protótipo de manipulador robótico capaz de coletar e mover pequenos objetos e que possa ser acoplado a um BlueROV2.

1.2 Objetivos Específicos

Os objetivos específicos são:

- Identificar os requisitos mínimos para a operação subaquática de robôs;
- Desenvolver um manipulador capaz de coletar pequenos artefatos;
- Estudar formas de proteção de motores elétricos contra o ambiente subaquático;
- Desenvolver um meio de comunicar dois dispositivos utilizando o umbilical do BlueROV2.

2 REVISÃO TEÓRICA

2.1 Robótica

O termo robótica tem sido utilizado ao longo da história da humanidade para descrever principalmente, máquinas que se assemelham aos seres humanos e que na imaginação de muitos, será a responsável pelo extermínio da humanidade, assim com nos contos e filmes de ficção científica. Contudo, a robótica possui características e objetivos diferentes ou até mesmo antagônicos a esta definição errônea. A palavra robô foi utilizada pela primeira vez no ano de 1920, pelo escritor tcheco Kareal Capek em sua peça chamada “Rossum’s Universal Robots”. Esta peça retrata um cenário fictício onde robôs humanoides dotados de sentimentos, inicialmente eram felizes ao realizar qualquer tipo de trabalho para os seres humanos, mas que em determinado momento se revoltam contra seus criadores e começam a almejar a extinção da raça humana (ROBERTS, 2016). Esta obra, deixando de lado o fato de demonizar os robôs, atrelou o termo robô a uma máquina que pudesse realizar trabalhos humanos, os quis geralmente levavam o trabalhador a fadiga. Este conceito informal se aproxima muito do conceito utilizado nos dias de hoje, como pode ser visto na tradução livre da definição da palavra robô utilizada pela Organização Internacional de Normalização - ISO (2012): "Robô - Mecanismo atuado programável em dois ou mais eixos com um grau de autonomia, movimentando-se dentro de seu ambiente, para realizar as tarefas pretendidas". Como pode-se notar nesta definição, um robô não precisa ter necessariamente o formato de um humanoide. A robótica pode ser dividida em duas grandes áreas: robótica móvel e manipuladores .

A robótica móvel compreende os robôs utilizados para movimentações, podendo ainda ser dividida em robôs terrestres, aéreos e subaquáticos. Um exemplo de robô móvel é o ROV, o qual é utilizado no ambiente subaquático para inspeções e explorações.

Os manipuladores robóticos são robôs que permanecem fixos em seu local de trabalho e geralmente executam atividades de manipulação de ferramentas ou objetos. Assim como na área da robótica móvel, os manipuladores podem ser divididos em sub-grupos sendo eles os manipuladores seriais, paralelos e híbridos. Um exemplo de manipulador robótico é o robô conhecido como braço antropomórfico, o qual é classificado como um manipulador serial é utilizado principalmente na indústria automotiva nas linhas de montagem de veículos.

2.1.1 Manipuladores Robóticos

Segundo Siciliano e Sciavicco (2010), manipuladores robóticos consistem em uma sequencia de corpos rígidos (elos) interconectados por articulações (juntas) que em conjunto com uma série de atuadores realizam determinadas operações conforme uma programação preestabelecida. O dispositivo responsável controlar o manipulador é geralmente um microcontrolador ou um microcomputador dedicado, o qual recebe comandos de entrada e converte estes comandos, segundo alguma lei de controle no acionamento dos atuadores presentes nas juntas, assim movendo os elos do manipulador de forma a executar o movimento solicitado pelo operador.

2.1.1.1 Cadeia Cinemática

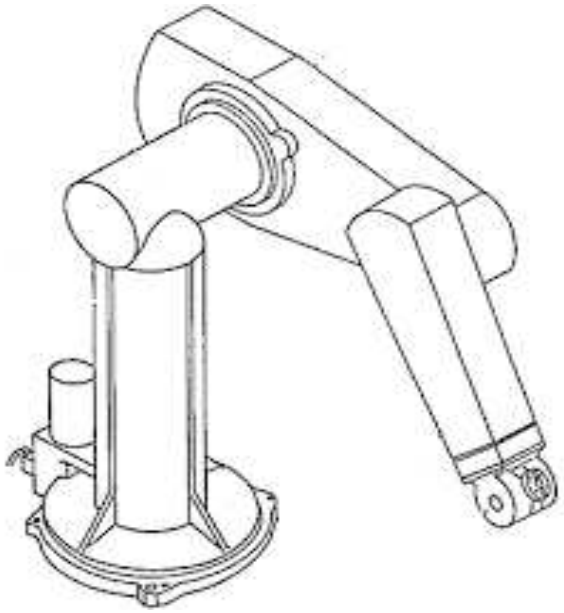
Manipuladores robóticos podem ser classificados, segundo sua cadeia cinemática de três formas: robôs seriais, paralelos ou híbridos. Segundo Antonelli e Khatib (2008) robôs seriais ou de cadeia aberta, possuem uma cadeia de elos e juntas a qual inicia na base e termina no efetuador final sem fechar em nenhum momento, tal configuração é ótima em aplicações as quais requerem um espaço de trabalho amplo e cargas de peso leve à mediano. Robôs paralelos ou de cadeia fechada, possuem mais de um conjunto de elos e juntas ligados ao efetuador final, provendo uma alta velocidade e precisão ao manipulador, mas em compensação possuem um espaço de trabalho reduzido em relação aos robôs seriais. Os robôs híbridos possuem características de robôs serias e paralelos sendo a configuração mais comum um manipulador com o fim da cadeia no formato de um robô serial e a base fechada como nos robôs paralelos. A figura 2 exibe um exemplo de manipulador robótico de cadeia aberta.

2.1.1.2 Elos e Juntas

Tsai (1999) diz que um manipulador é constituído por uma série de elos ligados por juntas e que o número de graus de liberdade (GdL) deste manipulador depende, além da quantidade de elos e juntas, do tipo de junta utilizada.

Os GdL de um manipulador são os movimentos únicos que o manipulador pode realizar. Por exemplo se um manipulador pode fazer apenas deslocamento no plano, ele possui 2 GdL sendo um no eixo x outro no y. se o mesmo manipulador agora puder fazer movimentos em z, ele possuirá 3 GdL. Esta configuração de 3 GdL translacionais caracteriza um robô do tipo cartesiano. Além dos GdL relacionados ao deslocamento, existem também os rotacionais que seguem o mesmo princípio. Os GdL rotacionais possuem nomes específicos sendo eles ângulos de euler *roll*, *pitch* e *yaw*. Estes termos são muito utilizados na aeronáutica no quando se fala em controle de

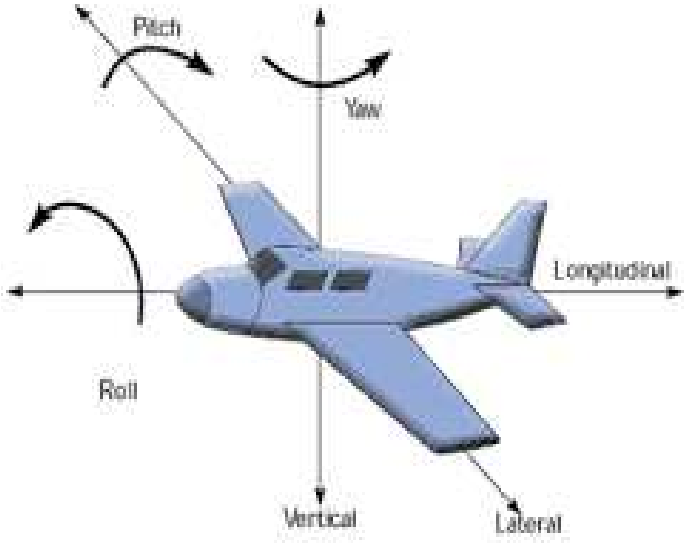
Figura 2 – Manipulador robótico de cadeia aberta.



Fonte: (SICILIANO; SCIAVICCO, 2010)

atitude da aeronave, tanto é que a figura 3 apresenta os GdL em questão, utilizando uma aeronave como exemplo.

Figura 3 – Ângulos de Euler.



Fonte: Novatel (2018)

Tsai (1999) também relata que os elos de um manipulador podem ser considerados corpos rígidos, ou seja, não sofrem efeitos deformação plástica, desde que não sejam empregadas altas cargas ou altas velocidades aos mesmos. Conforme relata Siciliano e Sciavicco (2010) a mobilidade de um manipulador é devida a presença

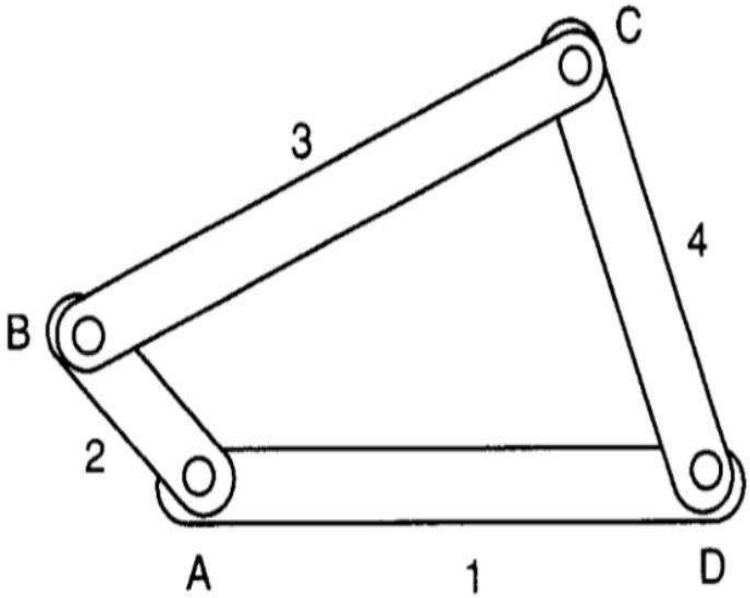
das juntas as quais podem ser rotativas, prismáticas ou uma combinação destas duas.

> Uma junta rotativa, R, permite 2 elementos pareados rotacionar um em relação ao outro, a partir de um eixo, o qual é definido pela geometria da junta. Consequentemente, a junta rotativa impõem 5 limitações ao par de elementos e é uma junta de 1 GdL. A junta rotativa é ás vezes chamada de par rotativo, par dobradiça ou junta pino. (TSAI, 1999, p. 7).

> Uma junta prismática, P, permite 2 elementos pareados deslizar um em relação ao outro ao longo de um eixo, o qual é definido pela geometria da junta. Consequentemente, a junta prismática impoem 5 limitações ao par de elementos e é uma junta de 1 GdL. A junta prismática é ás vezes chamada de par deslizante. (TSAI, 1999, p. 7).

A figura 4 mostra um mecanismo de 4 barras, no qual os elos são nomeados por números e as juntas rotativas nomeadas por letras.

Figura 4 – Mecanismo 4 barras



Fonte: (TSAI, 1999)

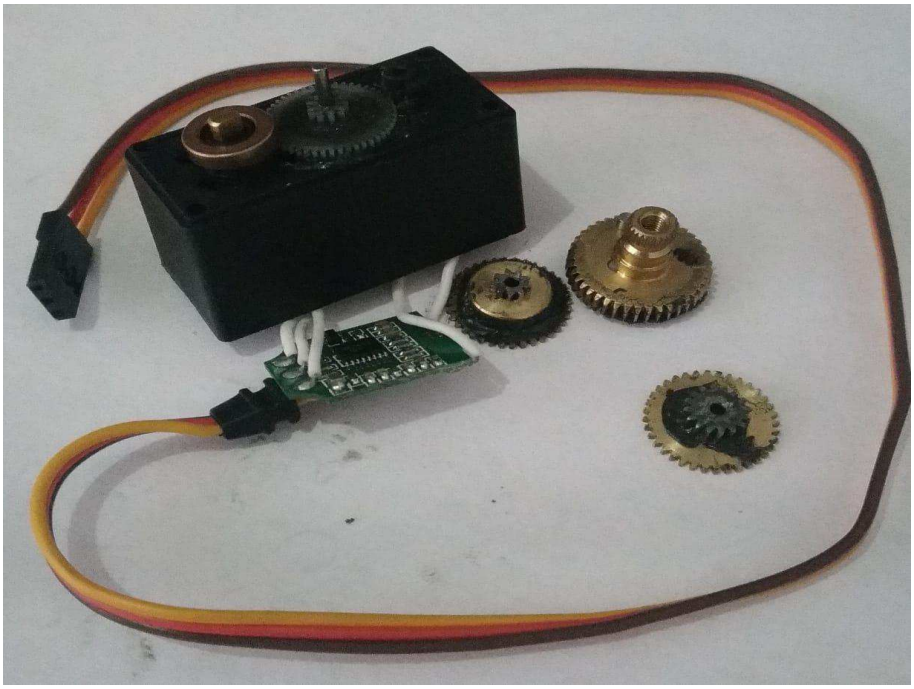
2.1.1.3 Atuadores

Segundo Christ e Wernli (2013, p. 511) "o mecanismo que transforma a força mecânica em movimento é o atuador". Ainda segundo Christ e Wernli (2013) atuadores podem ser dos tipos hidráulico, pneumático ou elétrico, onde os dois primeiros utilizam de conjunto bomba-pistão para gerar movimento, enquanto o último utiliza um motor elétrico para executar a mesma operação. O atuador do tipo elétrico é largamente utilizado em pequenos projetos por sua robustez e controle simplificado, mas normalmente é evitado em aplicações subaquáticas por ser sensível a este ambiente e em consequência disto ser necessário criar um isolamento mecânico para o mesmo.

2.1.1.3.1 Servomotor

Um dos atuadores elétricos mais utilizados em pequenos projetos é chamado servomotor. O servomotor é um componente eletromecânico que a partir de um nível de tensão de entrada, alinha o eixo do motor em uma determinada posição, a qual obrigatoriamente estará num intervalo máximo de 180 graus para este tipo de servomotor, uma vez que há uma limitador mecânico presente no sensor de posição do servomotor. O servomotor é constituído basicamente por um motor de corrente contínua convencional, um sensor para monitorar a posição do eixo do motor, um sistema de controle e uma caixa de engrenagens para realizar a ligação entre o eixo do motor e o sensor. A figura 5 mostra um servomotor desmontado.

Figura 5 – Servomotor desmontado.

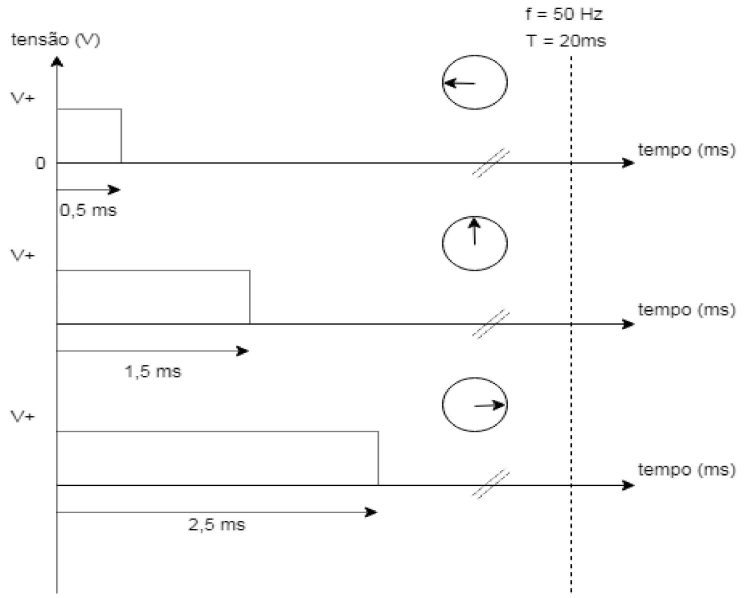


Fonte: AUTOR (2018])

A forma mais comum de controlar um servomotor deste tipo, é utilizar a modulação por largura de pulso (PWM), que nada mais é do que controlar a tensão média, modulando o tempo que a tensão máxima é aplicada. O servo motor possui três fios sendo um par utilizado para a tensão positiva , outro para a referência e o último sendo o sinal de controle, que neste caso é um sinal PWM. Geralmente o PWM é modulado numa frequência de 50 Hertz, o que define que um período de 20ms e a tensão máxima usual é de 5V. Contudo, geralmente para o controle do servomotor apenas uma faixa de 2 ms é utilizada, a qual encontra se entre 0,5 ms e 2,5 ms, para que a tensão média fique dentro dos valores de trabalho do controlador. Desta forma quando o sinal de controle permanece por apenas 0,5 ms o eixo do motor se alinha

com o angulo 0 graus, quando permanece por 1,5ms se alinha com o angulo de 90 graus e quando permanece por 2,5ms se alinha com o angulo de 180 graus. A figura 6 mostra um diagrama plotando a tensão versus o tempo, e um indicativo da orientação do eixo do motor.

Figura 6 – Diagrama tensão-tempo para controle do servomotor



Fonte: AUTOR (2018])

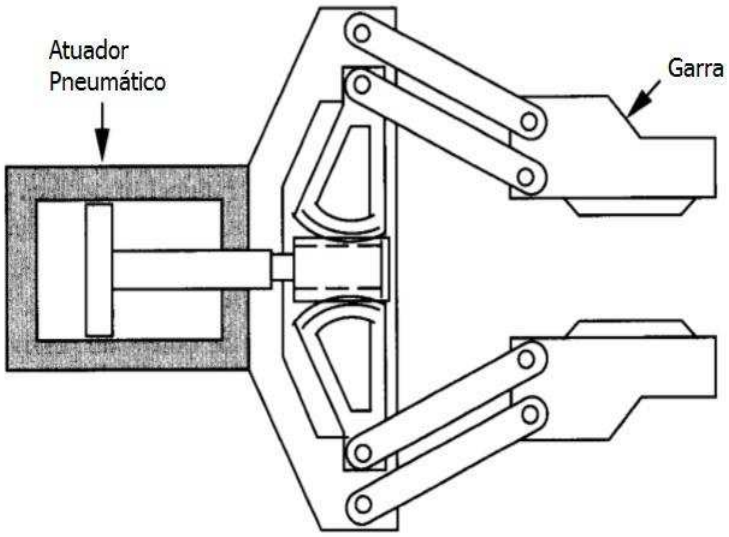
2.1.1.4 Efetuador Final

Um manipulador robótico tem como principal função mover objetos de uma posição a outra e portanto, necessita de uma ferramenta adequada para realizar tal trabalho, a qual na robótica é chamada de efetuador final.

> Efetuadores finais são dispositivos acoplados ao elo de saída de um manipulador mecânico para agarrar, levantar e manipular peças. Podemos pensar no efetuador final como uma interface mecânica entre o manipulador mecânico e o ambiente. (TSAI, 1999, p. 17).

A figura 7 mostra um exemplo de efetuador final do tipo pinça dupla acionado por um cilindro pneumático. Como a ação do efetuador final não é considerado um GdL, os fabricantes de manipulador utilizam o termo funções para representar a soma de todos os GdL com a ação do efetuador final, portanto um manipulador com n funções possui n - 1 GdL e a ação do efetuador final.

Figura 7 – Efetuador final do tipo pinça



Fonte: (TSAI, 1999)

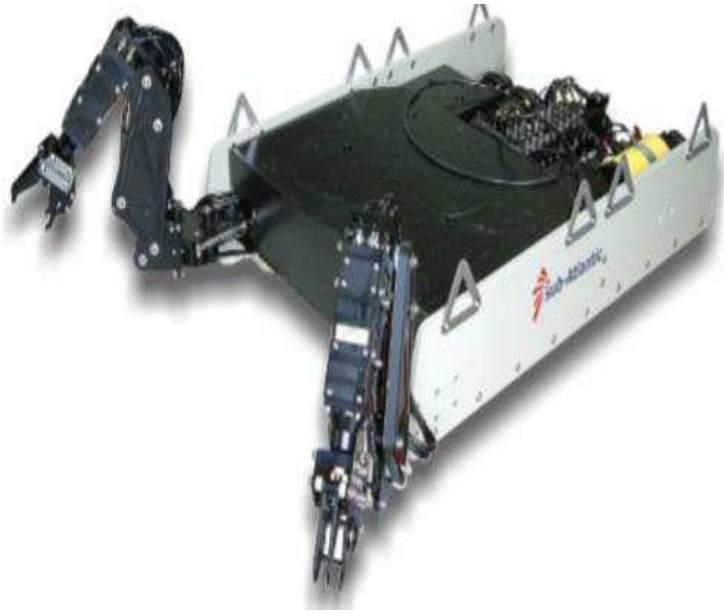
2.2 Robótica Subaquática

O entendimento científico do fundo do mar esta expandindo rapidamente nos últimos anos graças a robótica subaquática (ANTONELLI; KHATIB, 2008) uma vez que é possível realizar explorações em profunidades muito maiores utilizando robôs não tripulados. A robótica subaquática não tripulada de intervenção esta concentrada sobre os ROVs com manipuladores robóticos acoplados, este conjunto permite não apenas a exploração do ambiente subaquático mas também a intervenção no mesmo, como por exemplo na coleta de artefatos em profundidades elevadas. A figura 8 exibe um exemplo de ROV munido de 2 manipuladores robóticos com garras produzido pela empresa sub-Atlantic.

2.2.1 ROV

Antonelli (2006, p. 3) relata que o termo ROV "denota um veículo subaquático ligado fisicamente via cabo a um operador que pode estar em um submarino ou em um barco na superfície". Segundo Antonelli e Khatib (2008) o primeiro ROV, POODLE, foi costruído em 1953 com o objetivo de explorar o fundo do mar mediterrâneo. Entre as décadas de 1960 e 1970, os ROV's evoluíram para aplicações voltadas ao âmbito militar, como por exemplo, na recuperação de artefatos utilizados em treinamentos da marinha. Após a década de 1980 os ROVs estabeleceram-se como ferramentas para a indústria marítima na exploração de petróleo e mantenimento de dutos, bem como

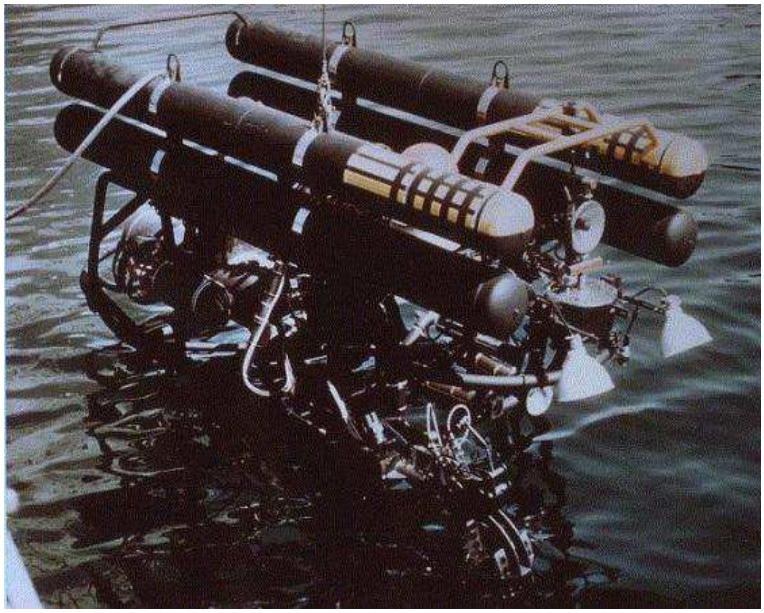
Figura 8 – ROV com 2 manipuladores robóticos da empresa sub-Atlantic



Fonte: (CHRIST; WERNLI, 2013)

para aplicações científicas como o estudo fauna e flora marítima. A figura 9 exibe o primeiro ROV desenvolvido pela marinha americana para a recuperação torpedos de teste chamado "Cable-Controlled Underwater Recovery Vehicle"(CURV).

Figura 9 – Cable-Controlled Underwater Recovery Vehicle (CURV)



Fonte: (USN, 2016)

Segundo Christ e Wernli (2013) ROVs são robôs não tripulados utilizados em diversas aplicações subaquáticas desde o suporte a mergulhadores até a construção subaquática e podem ser divididos segundo a combinação de seu peso e capacidades.

2.2.1.1 OCROV

Segundo Christ e Wernli (2013) os ROVs de observação mais conhecidos como OCROVs, possuem o menor peso dentre todas as classes possuindo um valor máximo de 100 kg e são constituídos basicamente por propulsores, bateria, câmera, controle, cabo e computador de bordo. Os propulsores são constituídos por motores de corrente contínua ligados a hélices, que o permitem alcançar profundidades de até 300m. Todo o sistema é alimentado por baterias, as quais se encontram junto a estrutura do robô bem como todos os outros componentes, salvo o controle. Esta classe não possui manipuladores robóticos acoplados, o que a restringe a operações de observação e suporte a mergulhadores. A figura 10 exibe uma coleção de OCROVs de variados tamanhos.

Figura 10 – OCROVs de variados tamanhos

Fonte: (CHRIST; WERNLI, 2013)

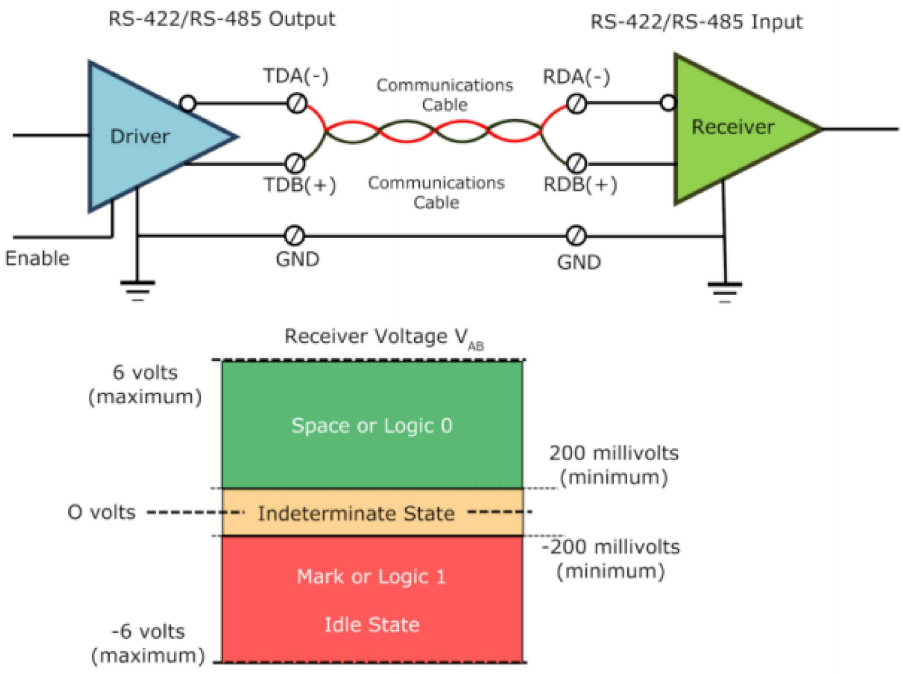
2.3 Padrão RS-485

Como citado por Salvi (2018), o padrão RS-485 é um padrão elétrico utilizado para implementar protocolos de comunicação serial. O padrão RS-485 é baseado na transmissão diferencial balanceada de sinais, através de um par de condutores, o qual é ideal para transmissão em altas velocidades, longas distâncias ou em ambientes propícios a interferência eletromagnética (SALVI, 2018). Este padrão também permite a comunicação entre vários elementos participantes em uma mesma rede e também permite a presença de mais de um mestre na rede.

A transmissão diferencial balanceada de sinais é caracterizada pelo envio de sinais de tensão de módulo igual mas com sinais opostos pelos dois condutores

independentes, como mostrado na figura 11.

Figura 11 – Transmissão Balanceada RS485



Fonte: (BB SMARTWORX, 2010)

Cada um destes sinais tem seu retorno realizado por um fio terra ou por um terceiro condutor de descarga, sendo a leitura do sinal pelo receptor, também realizada de forma diferencial. Segundo BB SMARTWORX (2010), receptores diferenciais balanceados medem a tensão entre os terminais A (-) e B (+), sendo que caso a tensão diferencial V_{AB} esteja entre os valores -200mV e -6V, o receptor interpreta o sinal como um nível lógico alto ou marca. Caso a tensão diferencial V_{AB} esteja entre os valores 200mV e 6V, o receptor interpreta o sinal como um nível lógico baixo ou espaço. Valores de tensão diferencial V_{AB} entre -200mV e 200mV, são considerados ruído e então último nível lógico é mantido. As características básicas do padrão RS-485 são:

- Distância máxima entre emissor/receptor de 1200 metros;
- Velocidade máxima de comunicação 10 Mbps;
- Tensão de alimentação do transmissor/receptor podendo variar entre -7V e 12V;
- Não obrigatoriedade de compartilhamento de sinal de referência entre dispositivos.

Segundo BB SMARTWORX (2010), o padrão RS-485 é muito utilizado principalmente na indústria, pois implementa apenas a camada física de comunicação deixando a cargo do usuário ou do fabricante de máquinas utilizar o protocolo que lhe convir. Alguns exemplos destes protocolos são o Profibus e Interbus. Contudo, esta liberdade de utilizar qualquer protocolo, muitas causa dificuldades ao tentar comunicar aparelhos de diferentes fabricantes.

3 MATERIAIS E MÉTODOS

Este trabalho teve como objetivo desenvolver um protótipo de manipulador robótico subaquático que pudesse ser acoplado ao BlueROV2 da UFSC Campus Joinville. O desenvolvimento do protótipo deu-se inicialmente por uma etapa de estudo e definição dos requisitos de projeto, bem como as condições do ambiente dos testes. Desta forma a construção do protótipo deu-se, quando necessário, pelas atividades de projeto, aquisição, construção e testes, dentro de cada um dos tópicos a seguir: Arquitetura do Manipulador, Arquitetura de Controle e Estrutura.

3.1 Requisitos

Inicialmente se fez necessário especificar qual seria a atividade ou quais conceitos o protótipo de manipulador deveria ser capaz de satisfazer. Com base nestas informações, foi possível relacionar os requisitos de projeto, os quais norteiam o desenvolvimento do protótipo.

Como este trabalho trata-se de um protótipo e é o primeiro nesta na área na UFSC Campus Joinville, optou-se por utilizar um ambiente controlado e de fácil visualização para a realização dos testes funcionais do manipulador, sendo então escolhida uma piscina convencional com profundidade aproximada de 1,8 metros, a qual é a profundidade média das piscinas residencias. Mesmo que no ambiente controlado não haja correnteza, deve-se levar em consideração este tipo de força, durante a etapa do projeto estrutural.

Definido o ambiente em que os testes funcionais seriam realizados, era necessário criar uma atividade que o manipulador juntamente ao BlueROV2 pudesse realizar. Um dos objetivos específicos deste trabalho é que o protótipo seja capaz de manipular pequenos objetos, portanto, a missão deste protótipo obrigatoriamente, deveria ser a realizar a manipulação de pequenos objetos no interior do ambiente citado anteriormente. Como não foi encontrado uma norma sobre testes de manipuladores subaquáticos, a massa e o tamanho dos objetos foi definido arbitrariamente como 500 gramas e cubos com lado 3 cm, respectivamente. É válido lembrar que o BlueROV2 possui uma câmera integrada que fica localizada na parte frontal do *skid* superior, e tal manipulação deve ocorrer dentro do campo de visão da câmera. Sendo assim o alcance e espaço de trabalho devem ser priorizados sobre a precisão. A cota entre o centro da câmera até a base do *skid* auxiliar é de aproximadamente 30 cm. Como o

manipulador será montado abaixo da câmera, esperasse que no mínimo o manipulador possua o alcance igual a esta cota, pois se houverem problemas com o movimento da câmera, ainda sim o manipulador poderá ser visto, caso a câmera esteja na posição perpendicular ao BlueROV2, a qual é sua posição inicial.

Ainda tratando sobre a câmera BlueROV2, deve-se analisar o campo de visão que a mesma oferece. A câmera integrada possui um grau de liberdade, sendo este uma rotação em *pitch* com limites de 90 graus para ambos os lados a partir da posição perpendicular ao BlueROV2. O fabricante também informa que a mesma possui uma abertura do campo de visão lateral de 110 graus (ROBOTICS, 2018). Com esta informação, é seguro afirmar que seria de grande valia se o manipulador pudesse alcance pontos por todo este campo de visão. Também é muito comum que o efetuador final do manipulador possa alcançar um determinado ponto em mais de uma configuração uma vez que não se pode prever todos os obstáculos que ambiente subaquático pode oferecer. Contudo, não é necessário que o manipulador possua mais do que um posição alternativa, uma vez o próprio BlueROV2 pode ajudar utilizando seus próprios graus de liberdade que no caso são 2 deslocamentos horizontais, um deslocamento vertical e uma rotação em *yaw* (ROBOTICS, 2018).

É importante levar em consideração que o manipulador é um *payload* do BlueROV2 e que no futuro este possa ser substituído por outro, como por exemplo um dispositivo para a medição dos níveis de poluição na água. Assim, é muito interessante que a sistema não seja projetado apenas para este tipo de *payload*.

O manipulador necessita de um periférico para a inserção de dados no sistema e este obrigatoriamente fica na superfície, junto a estação de base. O dispositivo utilizado deve ter entradas suficientes para movimentar cada uma das juntas do manipulador afim de provar o correto funcionamento de cada uma das juntas.

A comunicação entre o ROV e a base é um grande desafio em qualquer projeto deste tipo, uma vez que uma das poucas formas de enviar dados com segurança e velocidade é utilizar um meio físico. Felizmente o BlueROV2 utiliza um cabo umbilical de 100 metros de comprimento, o qual possui 4 pares de condutores trançados, sendo que apenas 1 destes pares é utilizado. Desta forma a solução mais adequada é a de realizar a comunicação utilizando 1 ou mais destes pares de condutores.

O manipulador deve ser acoplado a uma estrutura auxiliar montada abaixo do *skid* principal, para que, caso a missão do BlueROV2 não necessite do manipulador ou de outro tipo de *payload*, o mesmo possa ser retirado. Portanto, todos os componentes eletroeletrônicos devem ficar contidos dentro do invólucro auxiliar, sendo este fixado no *skid* auxiliar. A única ligação entre o ROV e o manipulador deve ser os condutores para comunicação citados acima.

O fornecimento de energia do BlueROV2 é feito através de uma bateria a qual esta embarcada em um dos invólucros principais. Como o manipulador é um sistema a

parte do BlueROV2, é interessante que o mesmo não compartilhe a bateria do sistema principal e possua sua própria, pois retirar o *payload* quando necessário se torna uma atividade penosa. Também não é interessante alterar a carga consumida pela bateria pois a mesma foi especificada apenas para o sistema do BlueROV2 de propulsão e controle. De toda forma, o manipulador deve utilizar soluções de baixo consumo de energia, uma vez que a autonomia da bateria utilizada decresce mediante ao aumento do consumo de carga.

A vedação é o requisito mais importante para a execução dos testes no ambiente controlado, com isso se faz de grande importância o estudo de técnicas de vedação uma vez que se tenha a arquitetura do manipulador congelada.

Por fim, como este é um protótipo, os custos do projeto devem se manter no mínimo possível, portanto sempre que possível, a utilização de tecnologia *open-source* ao invés da proprietária e a modificação de materiais materiais reaproveitados ao invés de materiais novos é valida, desde que seu desempenho seja similar a referência.

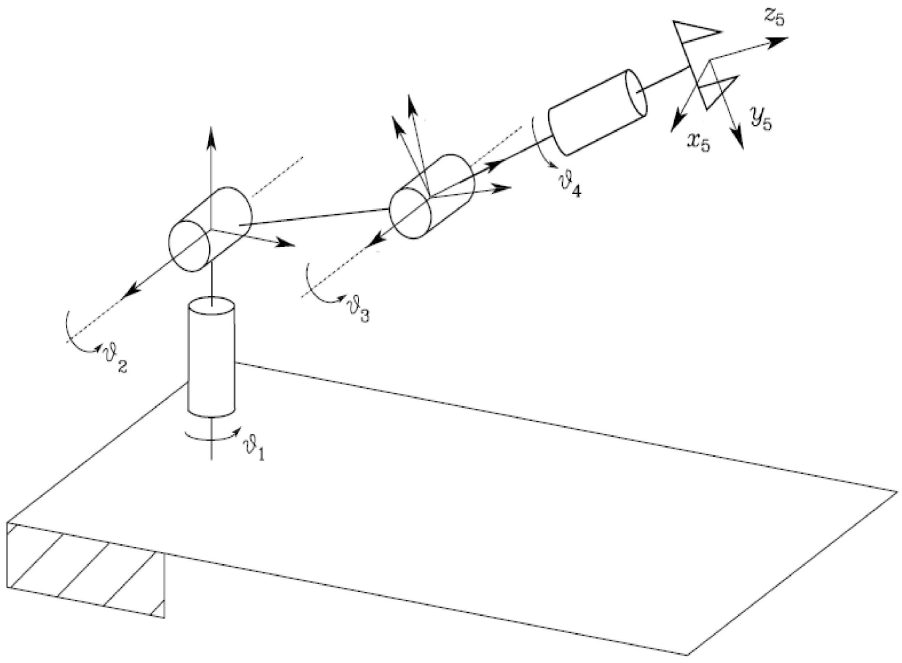
Com base nesta descrição informal da operação do manipulador em conjunto com o BlueROV2, foram relacionados os seguintes requisitos formais do projeto:

- Operar a uma profundidade de 2 metros em um ambiente subaquático controlado;
- Apanhar e manipular objetos cúbicos de lado 3 cm e massa 500 gramas;
- Possuir alcance mínimo de 30 cm;
- Espaço de trabalho lateral com abertura de 110 graus;
- Robustez para suportar forças externas provenientes de correntezas;
- Comunicação entre base e controlador utilizando condutores do umbilical;
- Desenvolver comunicação confiável através de meio físico com 100 metros de comprimento;
- Graus de liberdade suficientes para gerar mais de uma configuração das juntas do manipulador para um mesmo ponto;
- Não compartilhar a bateira do sistema principal;
- Possuir um periférico para controlar o manipulador;
- Controle junta a junta para testes.
- Utilizar tecnologia *open-source* quando possível;
- Buscar soluções alternativas antes de realizar a compra de materiais;
- Projetar o sistema de controle levando em conta que o manipulador possa ser trocado por outro *payload*;
- Minimizar e otimizar os componentes consumidores de energia elétrica, afim de maximizar a autonomia da bateria do manipulador;

3.2 Arquitetura do Manipulador

Inicialmente foi estabelecido que seriam construídos dois manipuladores, um deles faria a manipulação dos objetos e o segundo manipulador seria responsável por fazer a ancoragem do ROV em estruturas. Após diversas discussões o manipulador de ancoragem foi retirado do projeto, pois a missão do ROV seria realizada num ambiente controlado sem correntes marítimas e sem estruturas para ancoragem. Em seguida foi determinado que o *design* do manipulador seguiria as características de um braço antropomórfico com punho esférico, contudo a construção do punho seria simplificada desconsiderando os movimentos angulares de *pitch* e *yaw*. Esta arquitetura de cadeia aberta foi escolhida pois demanda pouco espaço para fixação e possui uma ampla área de trabalho, a qual pode contemplar todo o campo de visão da câmera acoplada ao ROV no casulo superior. Tal configuração do manipulador somado aos graus de liberdade que o próprio ROV fornece com sua movimentação, permite a coleta e manipulação de qualquer objeto presente no campo de visão do ROV. Como objetivo do manipulador é realizar a coleta e manipulação de objetos, o efetuador final escolhido foi uma garra do tipo pinça. A figura 12 mostra o a arquitetura proposta do manipulador, sendo ela formada por 4 juntas rotativas e um atuador, desta forma resultando em um manipulador de 5 funções.

Figura 12 – Arquitetura proposta do manipulador.



Fonte: Adaptado de (SICILIANO; SCIAVICCO, 2010)

Definida toda a arquitetura do manipulador, os atuadores das juntas foram selecionados sendo mini servomotores. Esta tecnologia foi selecionada por seu

controle simplificado, o qual não necessita de nada além de um sinal de controle e a alimentação. As tecnologias que utilizam pneumática e hidráulica foram descartadas sumariamente, pois necessitam de uma fonte de fluído pressurizado, a qual o projeto não contempla. Entretanto, os servomotores são seriamente agredidos pela água. A busca por servomotores à prova d'água retornou apenas soluções as quais garantiam o funcionamento até 1 metro de profundidade no máximo, o que não atinge o resultado da missão esperado, entretanto existem relatos e tutorias de métodos utilizados para realizar a selagem desse tipo de motor, os quais serão apresentados mais a frente.

Determinado que seriam utilizados servomotores, se deu início ao processo de aquisição dos atuadores, conforme as especificações do manipulador os dois primeiros movimentos do manipulador seriam submetidos a maior carga. Por mais que o pelos requisitos do projeto o alcance mínimo do manipulador fosse de 30cm, optou-se por iniciar o projeto com um manipulador com alcance de 50 cm. Assim, utilizando os dados de uma haste de 50 cm ligada em uma das extremidades ao servo motor e na outra extremidade fixada uma massa de 500 g, o torque estático a que o servomotor esta submetido possui módulo de 2,5 Nm. Contudo, como não se conhece a dinâmica deste mecanismo no ambiente subaquático, optou-se por utilizar um torque 20% maior, sendo este então fiado em 3 Nm. Após diversas consultas verificou-se que servomotores com este torque não são comuns no mercado e então seria necessário um investimento maior, desta forma foram adquiridas 3 unidades do servomotor SAVOX 1272SG o qual possui um torque de 3 Newton metro quando alimentado à uma tensão de 7,4 volts e 2,3 Newton metro a 6 Volts. Este servomotor também possui engrenagens de aço (SAVOX, 2018), as quais são muito mais robustas e duráveis que as feitas de plástico. Para as demais juntas foram adquiridos 5 unidades do servo TOWER PRO MG995, o qual possui um torque aproximado de 1 Newton metro quando alimentado a 6 volts, mas diferente do atuador anterior suas engrenagens são plásticas (TOWERPRO, 2018). A figura 13 mostra os dois modelos de servomotores adquiridos para as juntas do manipulador.

3.3 Controle

Inicialmente nesta etapa, foi definido como a arquitetura do sistema de controle seria construída e duas possibilidades possibilidades foram levantadas, a primeira seria integrar o manipulador robótico ao sistema de controle do ROV e a segunda opção, criar um pacote externo que fosse independente do controlador do ROV. Inicialmente a primeira opção parecia mais viável, pois, o controle do ROV é feito utilizado um software open-source, ou seja, seria possível alterar o programa do mesmo conforme desejado afim de resolver o problema proposto. Entretanto, a segunda opção de criar um sistema

Figura 13 – Modelos de servo utilizados nas juntas.

Fonte: AUTOR (2018)

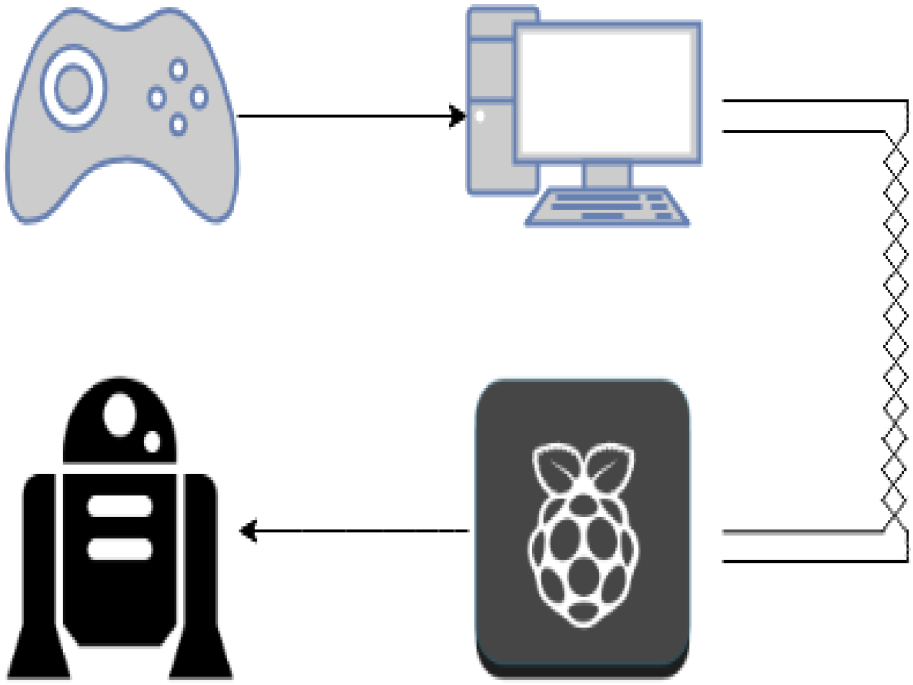
independente se fez mais viável, pois utilizando esta metodologia, seria possível agregar diversas outras funções ao ROV, uma vez que este módulo independente, sendo ele intercambiável, poderia ser alterado conforme a necessidade do usuário.

Assim, iniciou-se o projeto da arquitetura de controle do protótipo escolhendo quais dispositivos fariam parte do sistema, lembrando que todos os dispositivos embarcados seriam colocados dentro do casulo auxiliar, o qual é dedicado a este protótipo. A figura 14 mostra o desenho esquemático da arquitetura proposta, a qual é composta por um *joystick*, um computador emulando a estação de base, o meio físico para a comunicação, o controlador do manipulador e próprio manipulador.

O primeiro dispositivo selecionado foi o controlador embarcado do manipulador, um microcomputador RPi. Este dispositivo foi selecionado pois possui desempenho superior aos similares, possui sistema operacional e interface gráfica o que pode vir a ser útil para outros tipos de *payload*, *software open source* e possui diversas portas de comunicação de vários tipos, portanto pode receber diversos periféricos diferentes.

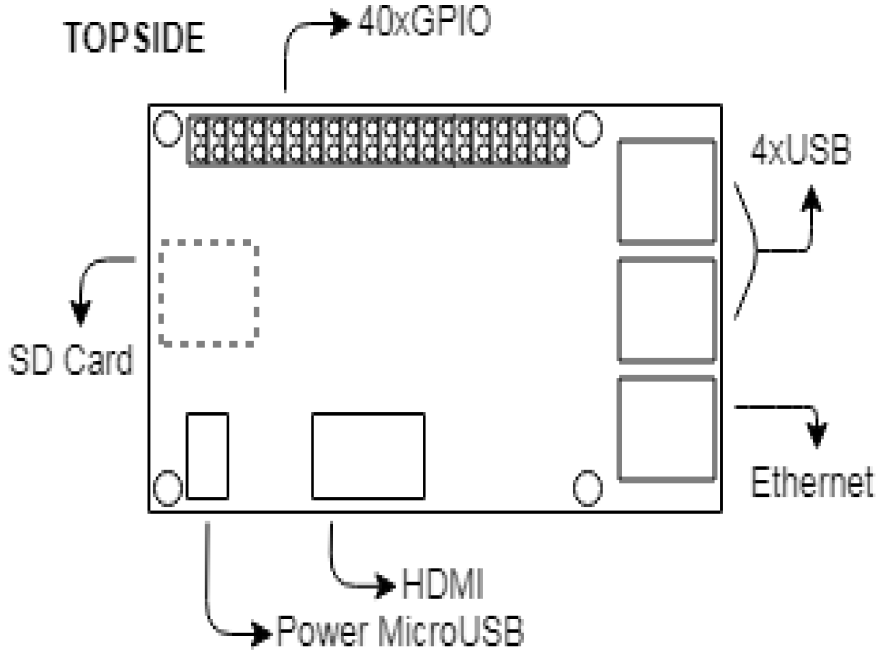
Assim foi adquirido um RPi 3 Modelo B, o qual possui um processador Quad Core 1.2 GHz, 1 GB de memória RAM, comunicação LAN, wireless e *Bluetooth*, 40 pinos GPIO, 4 portas USB e saída de vídeo HDMI (PI, 2018). A figura 15 mostra um desenho esquemático do microcomputador RPi destacando as componentes utilizados neste projeto.

Figura 14 – Modelo do sistema de controle.



Fonte: AUTOR (2018)

Figura 15 – Componentes utilizados no Raspberry Pi.



Fonte: AUTOR (2018)

Em posse do RPi se fez necessário instalar o sistema operacional Raspbian no cartão SD do dispositivo, tendo este a mesma função do disco rígido em um computador convencional. Inserido o cartão SD com sistema operacional em seu respectivo *slot*

e ligando o dispositivo via entrada micro USB, foram utilizados mouse USB, teclado USB e monitor HDMI afim de conectar o dispositivo a internet pela primeira vez para a realização das devidas atualizações e também para a configuração de um serviço de rede, utilizando o software VNC. A aplicação VNC é utilizado para realizar o acesso ao RPi remotamente, desta forma sendo necessário apenas um meio de comunicação entre um computador qualquer e o RPi, que neste caso foi escolhido como sendo uma comunicação cabeada utilizando a porta Ethernet.

Em paralelo a isto, a estação de base também foi definida, sendo ela seria composta apenas por um PC contendo a aplicação Python e um *joystick* conectado a uma porta USB, o qual seria responsável por controlar o manipulador. O *joystick* escolhido para este projeto foi um controle do console Xbox 360 USB genérico, pois o mesmo possui mais de 20 funções digitais e 6 funções analógicas. O critério para a escolha deste controle foi o de que ele era o mais barato, visto que o mesmo se encontrava avariado e alguns pontos de solda o fizeram voltar a funcionar. Para finalizar o sistema de controle, era necessário criar uma comunicação entre o sistema embarcado e a estação de base. Como o cabo umbilical do ROV - o qual é utilizado para a troca de informações entre o controlador do ROV e a estação de base, possuía veias livres, um par trançado foi utilizado para realizar a comunicação do protótipo via protocolo serial. Como não é possível conectar o par traçado diretamente aos dispositivos foram adquiridos 2 conversores USB para o padrão RS-485, os quais fazem a ponte entre os sistema embarcado e a estação de base, garantindo o envio e recepção das mensagens. Neste ponto todos os componentes do sistema de controle haviam sido adquiridos e então foi possível dar início aos testes de integração entre os dispositivos. A figura 16 mostra a montagem do sistema sem o manipulador afim de testar a recepção dos comandos provenientes do *joystick* e a comunicação serial entre o computador e o RPi, sendo estas funções implementadas por um algoritmo na linguagem Python utilizando as bibliotecas padrão e as bibliotecas de uso específico pyserial e inputs.

3.3.1 Integração do *joystick* utilizando a biblioteca inputs

Afim de realizar a comunicação entre o *joystick* e o computador, a biblioteca inputs do python foi utilizada, esta biblioteca foi utilizada pois ao contrário de pacotes mais conhecidos como pyQT e pygame, os quais fornecem uma interface gráfica amigável ao usuário final, esta biblioteca pode ser utilizada em segundo plano sem a necessidade de uma aplicação dedicada ou monitorada pelo usuário. Este módulo possui a capacidade de trabalhar com diversos periféricos como mouse, teclado e *joystick*, sendo que cada ação realizada em tais periféricos cria um evento, o qual pode ser consultado dentro do objeto criado para o periférico. A seguir é apresentado um

Figura 16 – Sistema de controle em bancada.

Fonte: AUTOR (2018)

trecho do código utilizado para testar módulo input:

```
1  import inputs
2
3  while(1):
4
5      events = inputs.get_gamepad()
6      print(events)
7      ##  [<inputs.InputEvent object at 0x0000000003017D30>]
8
9      for event in events:
10         print (event.ev_type, event.code, event.state)
11         ## Absolute ABS_RX 262
12         ## Key BTN_WEST 1
13         ## Sync SYN_REPORT 0
```

Na linha 1 a biblioteca é importada para dentro do código, fazendo com que todas as classes e métodos criados na biblioteca inputs possam ser acessados. Na linha 5 **events** é recebe uma lista de eventos a partir do método **inputs.get_gamepad()**, portanto toda vez que esta linha do código for executada **events** receberá uma lista dos eventos proveniente do periférico, neste caso os eventos podem ser um botão pressionado ou o movimento do *stick* analógico. Na linha 9 a a lista events é verificada e caso haja alguma informação contida nela, significa que algum evento ocorreu. Por

fim na linha 10 é impresso no console o tipo, o código e o estado do evento ou eventos ocorridos. Os eventos para este periférico podem ser do tipo *Absolute* que representa o acionamento de qualquer botão ou *stick* direcional, *Key* que representa o acionamento de qualquer botão de ação ou gatilho e *Sync* o qual é enviado antes e depois de cada um dos eventos citados acima afim de prover uma espécie de *handshake* assegurando o sincronismo no envio de informações.

Como esta biblioteca é orientada por eventos, a informação de que por exemplo um botão foi pressionado é enviada somente uma vez, onde neste caso o evento teria o estado igual a 1, contudo, se este botão permanecer pressionado o evento não será gerado novamente pois ocorre apenas nas bordas de subida e descida. Para sanar este problema foi desenvolvido um código simples de *Set Reset*, para que quando haja uma borda de subida do botão uma variável receba *TRUE* e somente receba *FALSE* quando o botão for liberado. Desta dorma foram mapeados os quatro botões direcionais, os quatro botões de açaõ da direita, os botões *START e BACK* e os dois botões superiores frontais, nenhuma das quatro entradas analógicas foi utilizada, pois o controle utilizado apresentou avaria nestas entradas. A tabela 1 relaciona as entradas do *joystick* mapeadas com suas devidas informações de evento.

Tabela 1 – Eventos mapeados do *joystick*.

	ev_type	code	state
Seta Esquerda	Absolute	ABS_HAT0X	0 ou -1
Seta Direita	Absolute	ABS_HAT0X	0 ou 1
Seta Acima	Absolute	ABS_HAT0Y	0 ou -1
Seta Abaixo	Absolute	ABS_HAT0Y	0 ou 1
A	Key	BTN_SOUTH	0 ou 1
B	Key	BTN_EAST	0 ou 1
X	Key	BTN_WEST	0 ou 1
Y	Key	BTN_NORTH	0 ou 1
L1	Key	BTN_TL	0 ou 1
R1	Key	BTN_TR	0 ou 1
BACK	Key	BTN_SELECT	0 ou 1
START	Key	BTN_START	0 ou 1

Fonte: AUTOR (2018)

3.3.2 Comunicação serial utilizando padrão RS 485 e a biblioteca pyserial

Como dito anteriormente, uma das poucas formas de enviar informações de forma segura a um dispositivo submerso é utilizando um meio físico. Neste caso o meio físico será um par de fios trançados livres do cabo umbilical do ROV. Para implementar esta comunicação foi decidido utilizar a comunicação serial com o padrão RS-485 com as configurações *half-duplex* e unidirecional, sendo que em ambas as pontas há um

conversor USB RS485.

Para realizar tal comunicação foi necessário desenvolver um programa em Python, o qual interpretasse as informações enviadas e recebidas através de portas escolhidas no computador e no RPi. A seguir encontrasse o código utilizado no computador, simulando o envio de um conjunto de números inteiros através do barramento serial:

```
1  import serial
2  import struct
3
4  ser = serial.Serial()
5  ser.baudrate = 9600
6  ser.port = 'COM6'
7  ser.bytesize = serial.EIGHTBITS
8  ser.parity = serial.PARITY_NONE
9  ser.stopbits = serial.STOPBITS_ONE
10 ser.timeout = None
11
12 ser.open()
13
14 msg = struct.pack('BBBBBBB',1,2,3,4,5,6,7)
15
16 ser.write(msg)
17
18 ser.close()
```

Nas linhas 1 e 2 são importadas as bibliotecas pyserial e struct, as quais disponibilizam acesso as portas serias do dispositivo e a manipulação e conversão de dados para diferentes bases, respectivamente. Entre as linhas 4 e 10 um objeto do tipo Serial é criado e configurado, sendo definidos a velocidade da comunicação, o encapsulamento da mensagem, o nome da porta serial utilizada e se há algum timeout para o envio das mensagens. Portanto, a porta serial a configurada utiliza *baudrate* 9600, com 8 bits de dados, 1 bit de parada e nenhum bit de paridade, conectado a porta USB "COM6"(padrão *windows*) e as mensagens enviadas no barramento são enviadas assim que são colocadas no barramento, uma vez que o *timeout* é nulo. Na linha 12 a porta serial configurada é aberta, sendo possível colocar informações no barramento. Nas linhas 14 e 16 uma mensagem formada por números inteiros é transformada em conjunto de *bytes* e enviado ao barramento, respectivamente. Na linha 18 a porta serial é fechada, encerrando a conexão.

Na sequência é mostrado o código utilizado para testar a recepção das mensagens, implementado no RPi:

```
1  import serial
2  import struct
3
```

```
4  ser = serial.Serial()
5  ser.baudrate = 9600
6  ser.port = 'tty0'
7  ser.bytesize = serial.EIGHTBITS
8  ser.parity = serial.PARITY_NONE
9  ser.stopbits = serial.STOPBITS_ONE
10 ser.timeout = None
11
12 ser.open()
13
14 msg = struct.unpack('BBBBBBB',ser.read(7))
15 print(msg)
16 ##(1, 2, 3, 4, 5, 6, 7)
17
18 ser.close()
```

Como no código anterior, nas linhas 1 e 2, as bibliotecas necessárias são importadas para dentro do código. Também como no código anterior entre as linhas 4 e 10 um objeto do tipo Serial é criado e configurado, com a diferença que a porta selecionada é a "tty0"(padrão Linux). A diferença para o código anterior reside entre as linhas 12 e 18, onde a porta serial é aberta e é esperada uma mensagem de 7 bytes, a qual é manipulada de forma a mostrar os 7 números inteiros enviados pelo computador e por fim fechando a porta serial, encerrando a conexão.

3.4 Estrutura

Com a arquitetura e os atuadores do manipulador definidos, foi dado início ao projeto e construção do manipulador e de sua estrutura de apoio, bem como a junção do manipulador ao skid do BlueROV2.

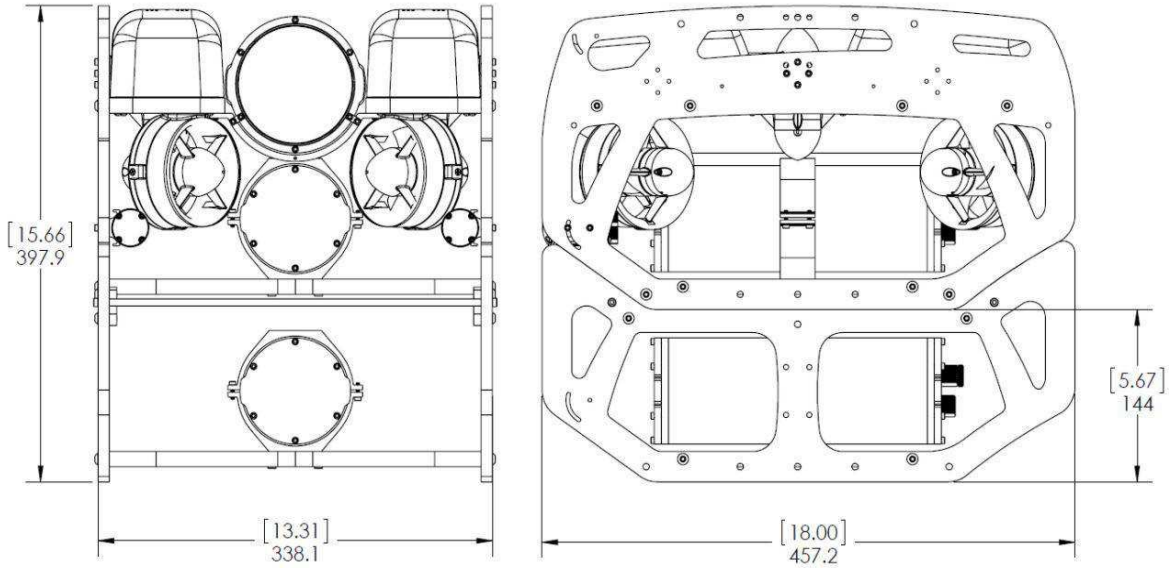
3.4.1 Skid e placa base

O BlueROV2 possui uma estrutura auxiliar denominada *skid* a qual permite o acoplamento de diferentes sistemas ao ROV e tem sua montagem realizada abaixo da estrutura do ROV, como mostrado na figura 18. Contudo, não mostrou-se viável acoplar um manipulador serial diretamente ao skid, então uma placa de nylon foi desenvolvida afim de distanciar o manipulador da estrutura, a qual pudesse ser fixada na base do skid. A placa de nylon utilizada, foi fabricada com as dimensões de 450 milímetros de comprimento, 150 milímetros de largura , 10 milímetros de espessura e sua fixação ao skid foi realizada utilizando parafusos e porcas, aproveitando os furos pre-existentes na base do skid.

Figura 17 – Skid auxiliar do BluROV2.

Fonte: (ROBOTICS, 2018)

Figura 18 – Arquitetura proposta do manipulador.



Fonte: (ROBOTICS, 2018)

3.4.2 Junta da Base

A junta da base do manipulador foi desenvolvida sendo a junta mais robusta do manipulador, pois o arrasto causado pela movimentação do ROV na água nos elos do manipulador, seria concentrado principalmente nesta junta. Desta forma a estrutura da base foi projetada sendo composta por um disco deslizante fixado à placa da base através de dois suportes laterias e um eixo de dois diâmetros montado com

interferência no disco deslizante, o qual liga o primeiro atuador ao segundo. A figura 20 mostra a estrutura da junta da base montada.

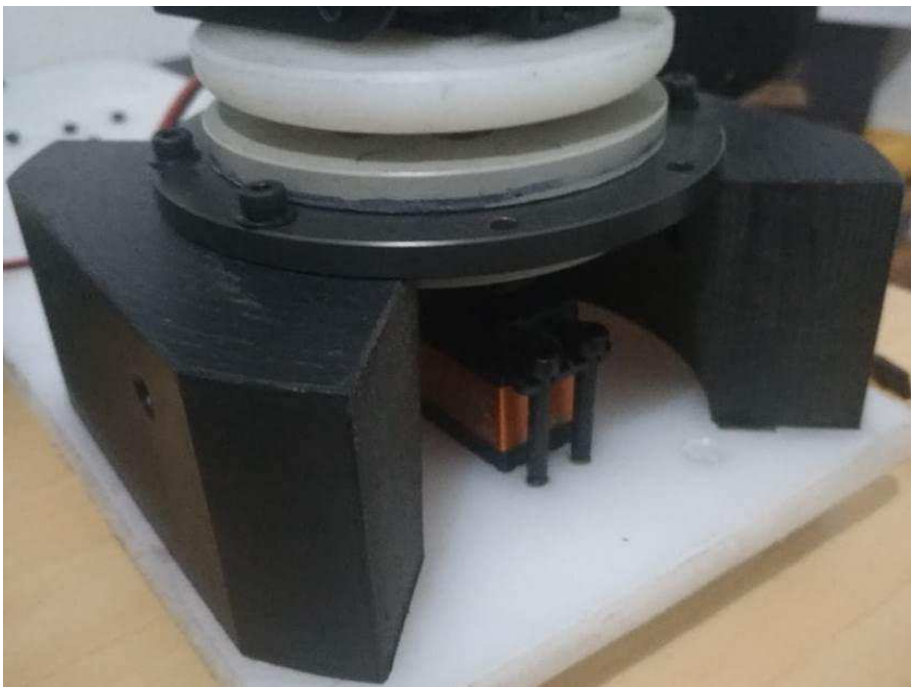
O disco deslizante como o mostrado na figura 19 é um elemento similar a um rolamento convencional, contudo suas características construtivas são diferentes. O disco deslizante não utiliza esferas e nem requer lubrificação, pois este é constituído por um anel externo o qual é fixado em alguma estrutura rígida e um disco interno que pode girar livremente, sendo o anel externo fabricado em aço inoxidável e o disco interno em um polímero especial. Tais características fizeram com que fosse escolhido utilizar este componente ao invés de um rolamento convencional, o qual necessitaria de uma chapa em que o rolamento pudesse ser montado com interferência.

Figura 19 – Disco deslizante IGUS.

Fonte: (IGUS, 2018)

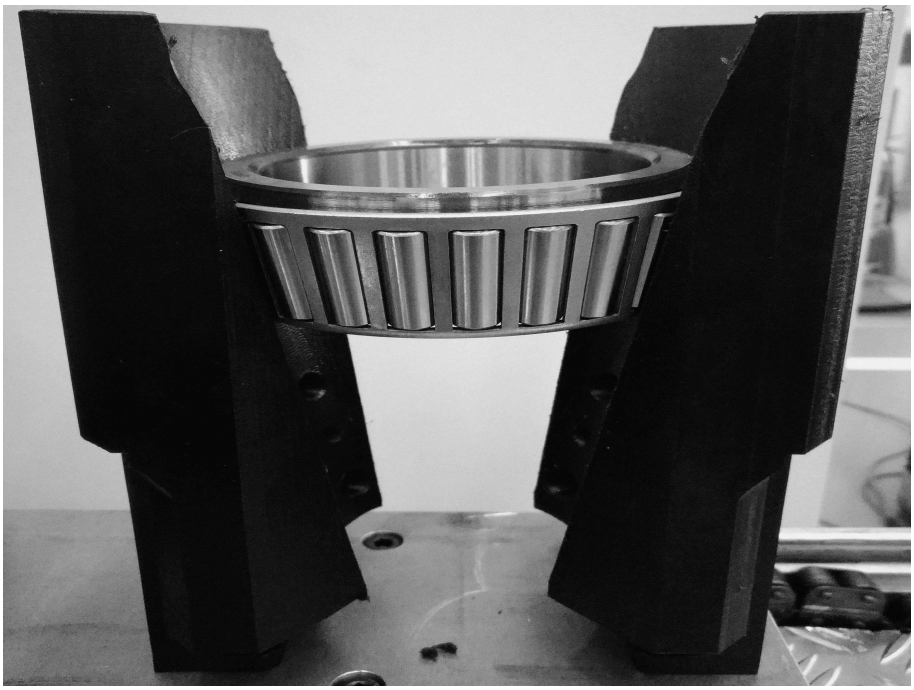
Os suportes utilizados para a fixação do anel deslizante à placa da base são feitos de nylon e possuem uma geometria complexa, a qual pode ser descrita como um sólido retangular, onde em uma de suas faces foi removido material seguindo as linhas de um cone. Estes elementos foram fabricados a partir de um conjunto o qual acomodava rolamentos cônicos de diversos tamanhos, o qual possuía uma altura inicial de 200 milímetros. A figura 21 mostra o conjunto citado acomodando um rolamento cônico, em sua função original. Como a peça original era muito mais alta que o necessário, foi utilizada uma fresadora CNC tridimensional para retirar material do topo do suporte, até que mesmo tivesse uma altura de 50 milímetros. A figura 20 mostra a junta da base montada e fixada à placa da base, sendo esta fixação realizada através de parafusos encontradas parte inferior.

Figura 20 – Estrutura da junta da base.



Fonte: AUTOR (2018)

Figura 21 – Função original do suporte da junta base.



Fonte: AUTOR (2018)

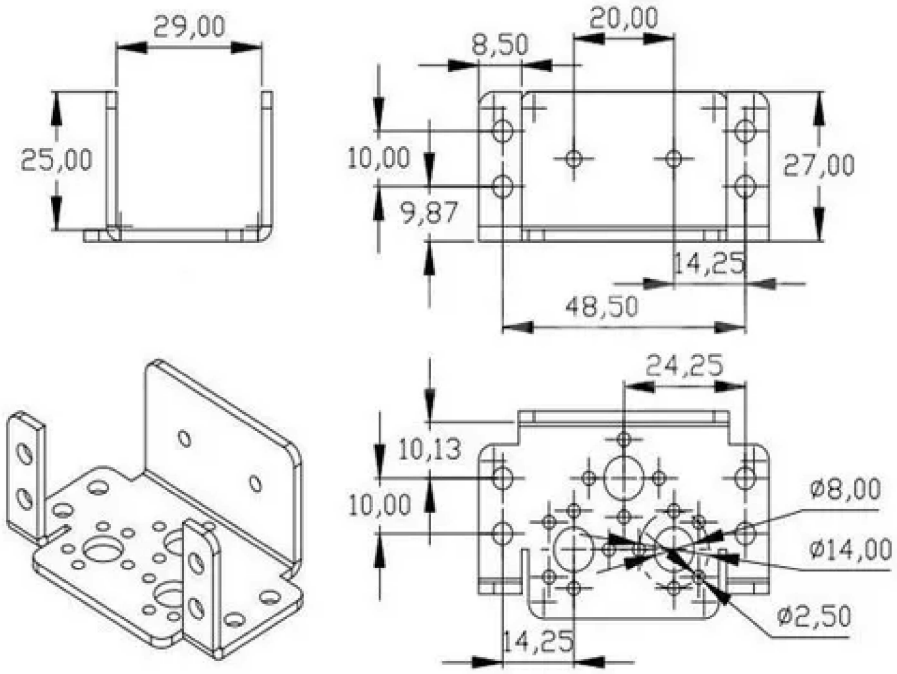
O eixo de dois diâmetro também foi reaproveitado de uma peça existente, mas este teve de passar por um processo de embuchamento, pois esta peça era vazada no sentido transversal e o diâmetro do orifício era muito grande, não sendo possível fixar o eixo do servomotor da base nele. Para resolver este problema, foi aumentado

o diâmetro deste orifício e inserido com interferência um pequeno cilindro de nylon, o qual foi ainda selado com cola resistente a médio torque para que não gerasse falso torque.

3.4.3 Suporte dos atuadores

Como os atuadores escolhidos seriam servomotores, seria necessário alguma estrutura que os fixasse uns aos outros afim de formar a cadeia do manipulador. Inicialmente pensou-se em fabricar diversas estruturas com impressão 3D as quais se encaixassem perfeitamente aos servos, contudo, ao realizar pesquisas sobre materiais e o que já havia no mercado, descobriu-se que haviam disponíveis suportes multi-funcionais para servos em aço inox, alumínio e outros materiais não corrosivos. As figuras 22 e 23 mostram os desenhos dos suportes, sendo estes produzidos no tamanho padrão de servos e portanto, pode-se utilizar qualquer tipo de servo nestes suportes.A figura 24 mostra a montagem de um dos suportes multifuncionais.

Figura 22 – Base suporte multi-funcional para servos.

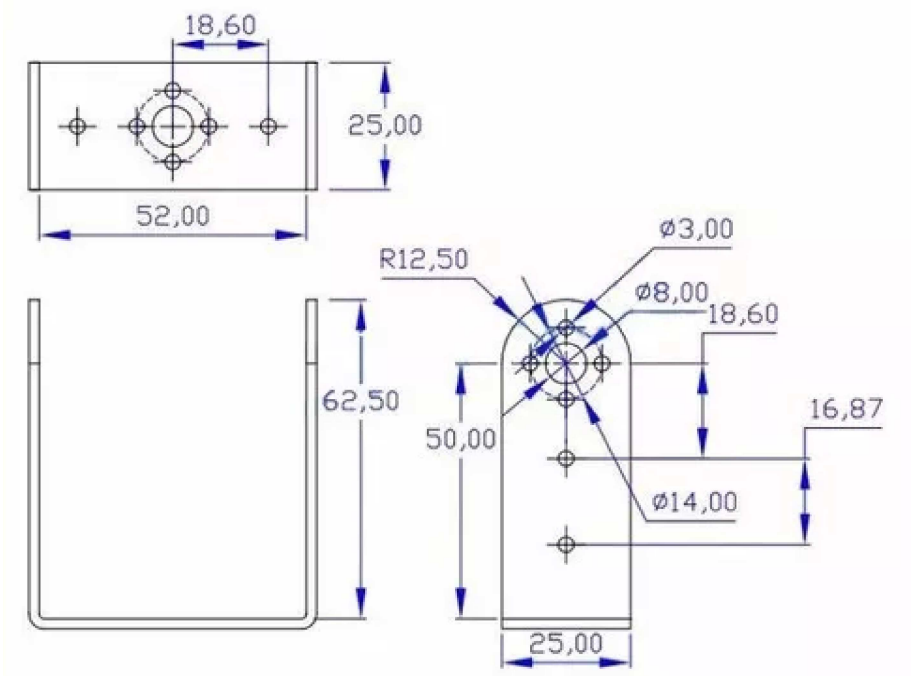


Fonte: MERCADO LIVRE (2018)

3.4.4 Distanciadores

Foram fabricados também 2 distanciadores em Nylon afim de deixar o manipulador com um alcance de 50 cm como definido anteriormente. Os distanciadores foram fabricados a partir de uma barra circular de uma polegada, sendo retirada dela dois pedaços de 10 cm cada. Após isto foram realizadas furações na extremidades

Figura 23 – Suporte em U multi-funcional para servos.



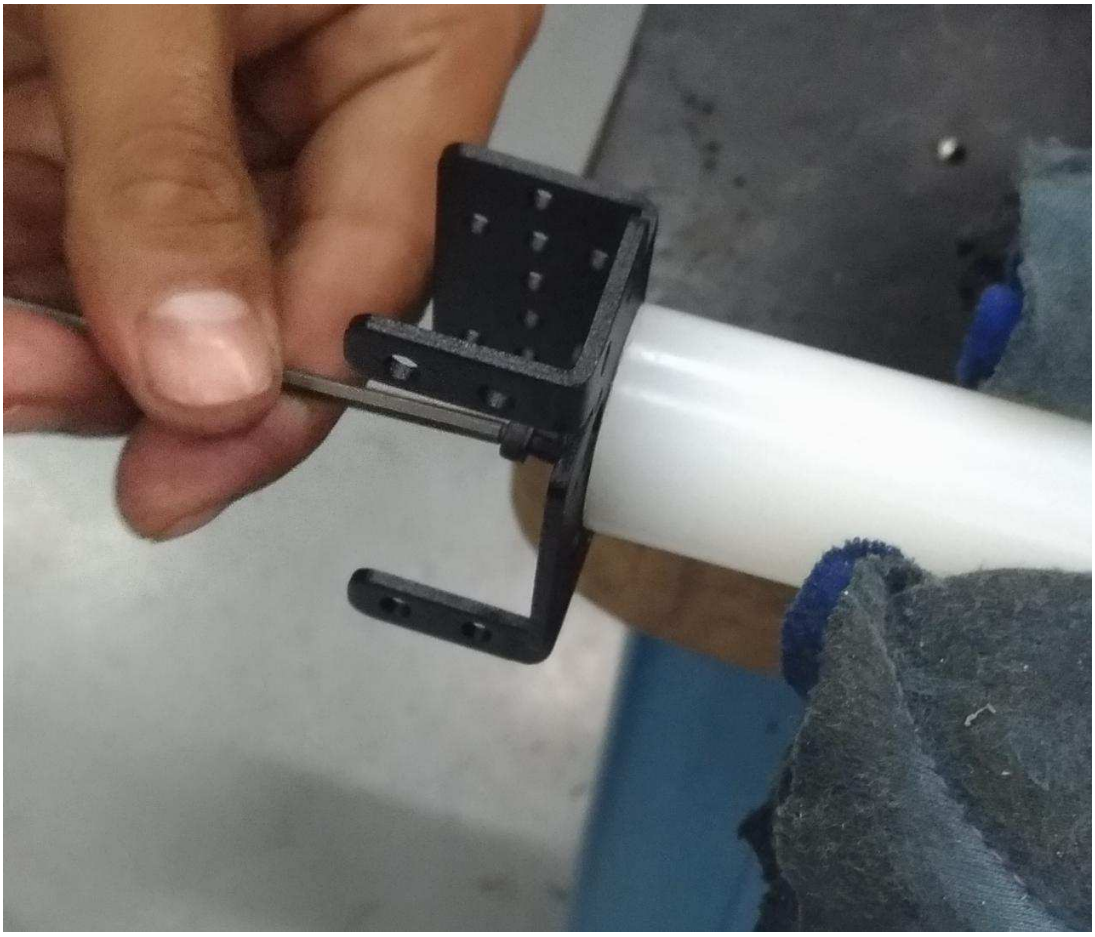
Fonte: MERCADO LIVRE (2018)

Figura 24 – Suporte multifuncional montado.

Fonte: AUTOR (2018)

para que pudessem ser fixados juntamente com os suportes multifuncionais. A figura

Figura 25 – Montagem da base multifuncional no distanciador fabricado.



Fonte: AUTOR (2018)

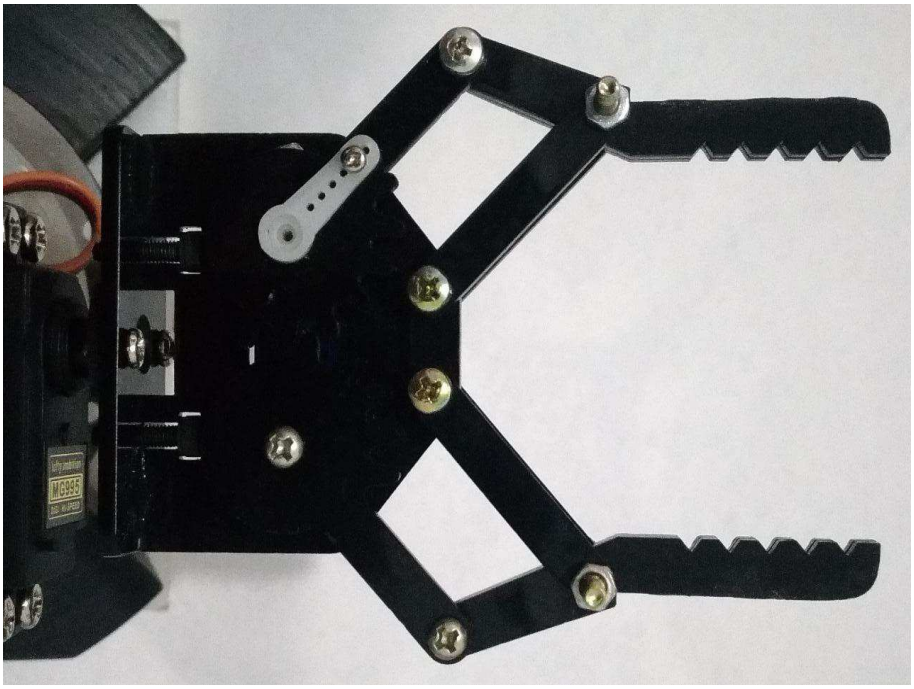
3.4.5 Garra

A garra adquirida para o projeto é uma garra plástica padrão para pequenos projetos de manipuladores e é mostrada na figura 26. Esta garra é formada por dois mecanismos 4 barras sendo um deles movido por um servomotor e o outro movido em sincronismo com o primeiro através de engrenagens e possui uma abertura máxima de 5cm, o que satisfaz o requisito de que o manipulador deve ser capaz de apanhar objetos cúbicos com 3 cm de lado. Não há um fabricante específico deste dispositivo, pois o projeto da mesma é simples e de fonte aberta.

Afim de testar a garra montada e os demais servomotores, o código abaixo foi desenvolvido com o intuito de controlar um servomotor através de um sinal PWM enviado pelo RPi. Segue abaixo o código de teste.

```
1 import pigpio
2 from time import sleep
3
```

Figura 26 – Garra robótica montada.



Fonte: AUTOR (2018)

```
4  pi= pigpio.pi()
5  pi.set_mode(17,pigpio.OUTPUT)
6  pi.set_PWM_frequency(17,50)
7  pi.set_servo_pulsewidth(17,1700)
8
9  while(pi.get_servo_pulsewidth(17) < 2000):
10     pi.set_servo_pulsewidth(17,pi.get_servo_pulsewidth(17)+5)
11     sleep(0.01)
12
13 while(pi.get_servo_pulsewidth(17) > 1000):
14     pi.set_servo_pulsewidth(17,pi.get_servo_pulsewidth(17) 5)
15     sleep(0.01)
16
17 pi.set_servo_pulsewidth(17,0)
18
19 pi.stop()
```

Nas linhas 1 e 2 são carregadas as bibliotecas para controle dos servos, sendo a biblioteca pigpio responsável por fazer a interface entre os pinos físicos do RPi e as memórias utilizadas no programa. Entre as linhas 4 e 7 o pino GPIO 17 é configurado como sendo uma saída PWM com frequência de 50 Hz e iniciando com a saída em nível alto por 1700 milissegundos. Entre as linhas 9 e 11, saída PWM é incrementada em 5 milissegundos através de um laço de repetição até que alcance a marca de 2000 milissegundos. O mesmo laço se repete entre as linhas 13 e 15, mas desta vez

decrementando 5 milissegundos até alcançar a marca de 1000 milissegundos. A partir da linha 17, o sinal PWM é desligado e o pino GPIO é liberado.

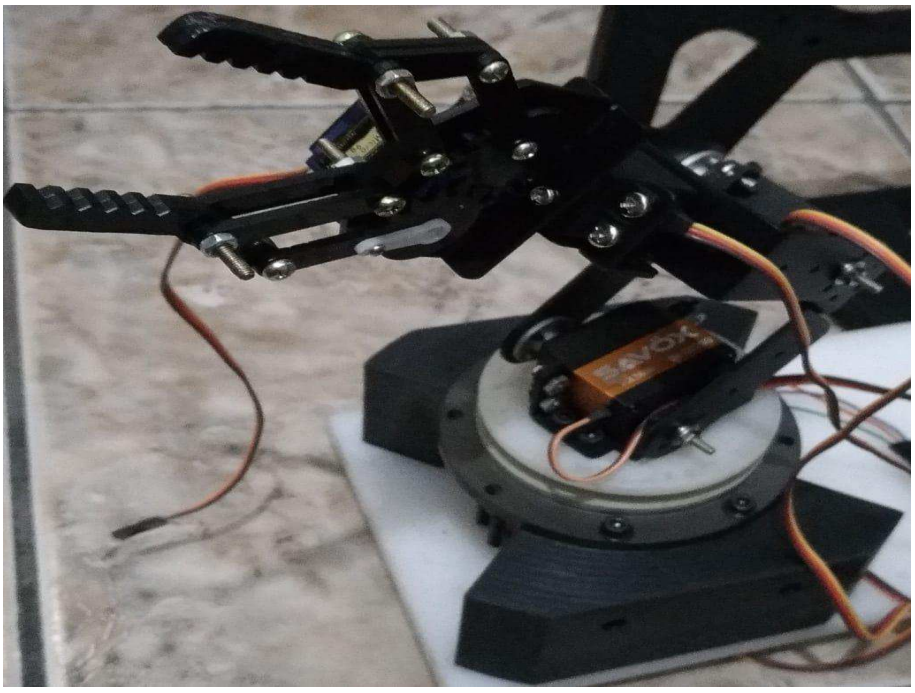
Durante os testes com os servomotores percebeu-se que o braço do manipulador com alcance de 50 cm causava uma imprecisão muito grande durante a manipulação. Portanto, foi decidido retirar os distanciadores fabricados, afim de deixar o manipulador com 30 cm de alcance. Esta alteração aumentou a precisão do manipulador e ainda deixou o projeto de acordo com os requisitos. Contudo, como os servo motores já haviam sido comprados e instalados a junta da base e do ombro ficaram sobre dimensionadas, pois como o braço de alavanca diminui o torque necessário também diminuiu. De toda forma isto não é um empecilho uma vez que o sobre dimensionamento do torque do motor não causa nenhuma consequência no desempenho do manipulador, causando apenas prejuízo financeiro pois poderia ser utilizada uma solução mais barata.

4 INTEGRAÇÃO, TESTES E VEDAÇÃO

4.1 Integração

Com todos os sistemas testados individualmente avançou-se para a etapa de integração onde, além da montagem dos componentes físicos, se fez necessário desenvolver aplicações customizadas na linguagem Python para o computador e para o Raspberry Pi. O manipulador totalmente montado pode ser visto nas figuras 27 e 28.

Figura 27 – Manipulador montado 01.



Fonte: AUTOR (2018)

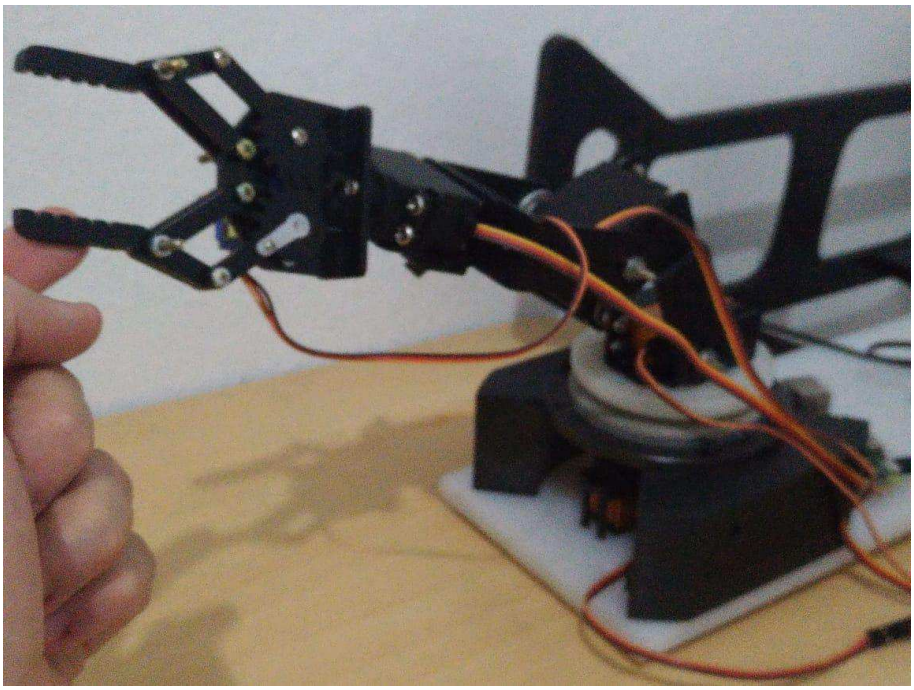
4.1.1 Aplicação do Computador

A aplicação executada no computador foi desenvolvida afim de monitorar os eventos criados pelo *joystick* e montar uma mensagem a qual é codificada em bytes e enviada para o Raspberry, toda vez que um evento é gerado pelo *joystick*.

4.1.1.1 Monitoramento de Eventos

Como dito anteriormente a biblioteca que implementa os métodos de interface entre o *joystick* e o computador é orientada a eventos, portanto não seria viável utilizar

Figura 28 – Manipulador montado 02.



Fonte: AUTOR (2018)

diretamente o retorno do método para enviar um comando ao Raspberry, pois o mesmo não é um valor contínuo apenas um pulso decorrente do evento de pressionar o botão. Desta forma, foi necessário desenvolver um código para monitorar os eventos de cada botão, atribuindo a uma variável um valor diferente de zero quando o botão é pressionado e 0 quando o botão estiver ocioso. O trecho de código abaixo mostra uma variável, a qual é alterada conforme o acionamento de dois botões.

```
1 if (event.code == 'BTN_EAST' and event.state == 1):
2     a = 1
3 elif (event.code == 'BTN_WEST' and event.state == 1):
4     a = 2
5 elif ((event.code == 'BTN_WEST' and event.state == 0) or
6       (event.code == 'BTN_EAST' and event.state == 0)):
7     a = 0
8 else:
9     a = a
```

Na linhas 1 e 2 é atribuído o valor 1 a variável caso o evento gerado tenha sido causado por pressionar o botão B do *joystick*. Nas linhas 3 e 4, caso a primeira condicional não seja verdadeira é verificado se o evento gerado é proveniente do acionamento do botão X do *joystick*, se sim, a variável recebe o valor 2. Nas linhas 5, 6 e 7, caso nenhuma das condicionais anteriores seja satisfeita e o evento gerado tenha sido causado por soltar qualquer um dos dois botões, é atribuído o valor 0 a variável.

Ainda sim, se nenhuma das concionais anteriores for satisfeita, significa que o evento gerado não possui relação com nenhum dos dois botões e portanto nas linhas 9 e 10, a variável mantém o último valor a ela atribuído.

4.1.1.2 Mensagem

A mensagem enviada do computador para o Raspberry é constituída por 7 números inteiros sendo que cada um desses inteiros possui uma função agregada e é alterado em função de um ou mais botões do *joystick*. O primeiro caractere da mensagem representa o botão BACK do *joystick* e tem a função de ligar ou desligar o sistema. Os caracteres entre as posições 2 e 6 da mensagem representam os botões laterais e superiores os quais são responsáveis pelas movimentações das juntas e da garra do manipulador. O último caractere representa o botão START o qual é responsável por inibir a movimentação de qualquer uma das juntas ou da garra quando ativo.

Os caracteres que responsáveis pela movimentação das juntas podem variar entre os valores 0, 1 e 2 sendo que se o valor atribuído ao caractere for zero a junta deve permanecer parada e se o valor atribuído for 1 ou 2 a junta deve se mover no sentido anti-horário ou horário, respectivamente. Já os caracteres que representam funções de controle do sistema podem variar entre valores 1 e 0, o que representa estado de ativado e desativado respectivamente. A tabela 2 mostra um resumo das características descritas acima.

Tabela 2 – Listagem das funções de cada caractere.

Nº do caractere	Botões	Função	Range
1	Back	Inicia/Encera a Aplicação	0, 1
2	B e X	Move Junta 1	0, 1, 2
3	A e Y	Move Junta 2	0, 1, 2
4	Cima e Baixo	Move Junta 3	0, 1, 2
5	Direita e Esquerda	Move Junta 4	0, 1, 2
6	R1 e L1	Abre/Fecha Garra	0, 1, 2
7	Start	Inibe Movimentação	0, 1

Fonte: AUTOR (2018)

4.1.2 Aplicação do Raspberry

A aplicação desenvolvida para o Raspberry tem como finalidade ler a mensagens colocadas no barramento serial, decodificá-las e realizar as ações por requeridas. Esta aplicação também é responsável por gerenciar a posição dos servomotores afim de tentar prevenir colisões.

4.1.2.1 Prevenção de Colisões

Uma das maiores preocupação quando se fala em manipuladores robóticos são as colisões, sejam elas com o próprio manipulador ou com ambiente em que o manipulador esta inserido. Durante a montagem foram verificadas diversas situações em que o manipulador poderia colidir e experimentalmente foram definidos os intervalos de angulação de cada servo motor.

Na junta 1, verificou-se que poderiam haver colisões com o SKID caso o servomotor chegasse próximo aos seus limites laterais. Portanto, o intervalo de trabalho desta junta foi definido como 45 graus a partir do centro em ambas as direções. Para a junta 2 foi escolhido um intervalo de trabalho de apenas 45 graus a partir do centro para fora, pois se o ângulo for menor que 90 graus à risco de colisão com o ROV e caso seja maior que 135 graus é possível que ocorra a colisão da junta 3 com a base. A junta 3 não recebeu nenhuma limitação em seu espaço de trabalho, pois, mesmo havendo risco de colisão, a mesma necessita trabalhar em toda a sua extensão para não diminuir a manobrabilidade do manipulador. A junta 4 não oferece risco de colisão uma vez que realiza o giro da garra no sentido axial. Por fim, o servomotor da garra teve seu movimento limitado na faixa de 0 a 45 graus pois case exceda o limite superior, o mecanismo de engrenagens que realiza o sincronismo entre as duas pinças é desencaixado. A tabela 3 mostra todas as informações citadas a cima e também faz a tradução do intervalo de ângulo para a largura de pulso.

Tabela 3 – Servomotores e seus limites.

Servomotor	Ang. Mínimo	Ang. Máximo	Largura de Pulso
1	45	135	1000 - 2000
2	90	135	1500 - 2000
3	0	180	500 - 2500
4	0	180	500 - 2500
5	0	45	500 - 1000

Fonte: AUTOR (2018)

4.1.2.2 Movimentação dos Servomotores

Em posse dos intervalos de ângulo de cada servomotor, duas funções foram implementadas para gerenciar a movimentação dos servomotores. Em ambas as funções, quando o método é invocado, a porta GPIO do Raspberry Pi - a qual foi configurada previamente, é passada como parâmetro bem como a largura de pulso limite máxima ou mínima do servomotor alvo, afim de realizar a movimentação angular requerida. O trecho de código abaixo mostra como os métodos citados foram implementados.

```
1 def angle_Up(GpIO,LH):
```

```
2       if(pi.get_servo_pulsewidth(GpIO) < LH):
3           pi.set_servo_pulsewidth(GpIO,pi.get_servo_pulsewidth(GpIO)+5)
4
5   def angle_Down(GpIO,LD):
6       if(pi.get_servo_pulsewidth(GpIO) > LD):
7           pi.set_servo_pulsewidth(GpIO,pi.get_servo_pulsewidth(GpIO) 5)
```

Ambos os métodos funcionam de forma similar, pois após invocados ambos verificam se a largura de pulso atual da saída PWM escolhida, esta contida dentro do intervalo de segurança que foi passado como parâmetro e caso o mesmo esteja contido neste intervalo, é realizado um incremento ou decremento de 5 unidades na largura de do pulso nesta saída PWM. Caso a largura de pulso atual não esteja no intervalo de segurança nada será feito, sendo possível apenas mover servomotor na outra direção.

4.2 Testes de Integração

Como este projeto se deu de uma forma muito prática, os testes de integração refletiram os resultados dos testes unitários, comprovando que o manipulador é funcional. Contudo, um ajuste em especial pode ser citado, sendo este a inserção de um *delay* entre a atualização da posição de cada servomotor. A aplicação desenvolvida no Raspberry Pi atualiza o valor de todas as saídas PWM ao mesmo tempo quando uma mensagem chega através do barramento serial, entretanto, quando um botão é pressionado no *joystick*, o respectivo caractere relacionado ao botão é alterado, fazendo com que a cada ciclo de *scan* do Raspberry, a largura de pulso do relacionada a este botão seja alterado. Em função disto, sem que houvesse um *delay* entre a atualização das posições dos servomotores, o servomotores se moviam numa velocidade alta, impedindo que houvesse qualquer destreza na manipulação e também acarretava em picos de reação no manipulador, que com o tempo poderiam causar avarias. A inserção deste *delay* na prática significa diminuir a taxa em que a posição varia, o que por consequência diminuiu a velocidade. Assim, de forma empírica, foi determinado que o *delay* utilizado seria de 10 milissegundos entre atualizações das posições dos servos.

Estes testes também mostraram que o consumo de corrente em vazio de todos os servomotores em vazio, varia entre 1 e 2 amperes, o que é uma informação valiosa no momento da escolha da bateria. Contudo, não se tem informação sobre como o consumo de corrente se daria no ambiente subaquático e por tanto seria necessário um teste prático antes de especificar a mesma.

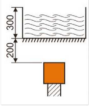
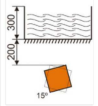
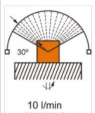
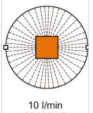
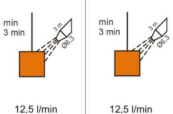
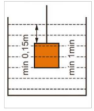
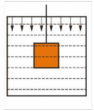
4.3 Vedação

O trabalho de realizar a proteção contra a água não foi realizado neste trabalho, impedindo que fosse possível realizar testes no ambiente subaquático. Contudo, um estudo sobre a vedação dos atuadores será realizado.

O primeiro passo quando se fala em vedação é ter em mente qual parte do manipulador necessita ser vedada ou selada. Como toda a eletrônica esta selada dentro do invólucro, os únicos elementos que necessitam de vedação são os motores elétricos, mais especificamente dos mini servomotores.

Todo fabricante de motor, seja ele qual tipo for, informa no *datasheet* do motor além de informações como tensão, corrente e torque, o grau de proteção sendo esta informação representada pela sigla *IP* seguida por 2 números quando este utilizar o padrão DIN ou *type* seguido de 2 ou 3 caracteres quando utilizar o padrão NEMA. Vamos aqui tratar apenas dos motores fabricados com padrão DIN, o qual é o padrão que segue as normas internacionais da IEC. A figura 29 mostra uma tabela contendo a relação entre os graus de proteção contra objetos sólidos e os graus de proteção contra água.

Figura 29 – Tabela de grau de proteção IP.

GRAU DE PROTEÇÃO CONTRA OBJETOS SÓLIDOS	PRIMEIRO NUMERAL	SEGUNDO NUMERAL / GRAU DE PROTEÇÃO CONTRA ÁGUA								
		0	1	2	3	4	5	6	7	8
		Não protegido	Protegido contra quedas verticais de gotas d'agua	Protegido contra quedas verticais de gotas d'agua em inclinação máxima de 15°	Protegido contra água aspergida de um ângulo de ± 69°	Protegido contra projeções de água	Protegido contra jatos de água	Protegido contra jatos potentes de água	Protegido contra imersão temporária	Protegido contra submersão
										
					10 l/min 80 kN/m2	10 l/min 80 kN/m2	12,5 l/min 30 kN/m2	12,5 l/min 30 kN/m2		
	Não protegido	0	IP00	IP01	IP02					
	Protegido contra objetos sólidos com Ø maior que 50 mm	1	IP10	IP11	IP12					
	Protegido contra objetos sólidos com Ø maior que 12 mm	2	IP20	IP21	IP22	IP23				
	Protegido contra objetos sólidos com Ø maior que 2.5 mm	3	IP30	IP31	IP32	IP33	IP34			
	Protegido contra objetos sólidos com Ø maior que 1 mm	4	IP40	IP41	IP42	IP43	IP44	IP45	IP46	
	Protegido contrapoeira depressão: 200 mm de coluna d'água. Máxima aspiração de ar: 80 x o volume do invólucro	5					IP54	IP55	IP56	
	Totalmente protegido contra a poeira. Mesmo procedimento de teste	6						IP65	IP66	IP67	IP68

Fonte: (OPUS, 2015)

A maioria dos mini servomotores tem grau de proteção 4 ou 5, pois toda a eletrônica e o motor estão envolvidos pela caixaria do servo. Contudo, para a aplicação subaquática seria necessário que os mini servomotores tivessem grau de proteção 8 contra água, mas infelizmente, estes ainda não existem no mercado. Quando se fala em servomotores maiores, existem produtos no mercado que possuem o grau de proteção adequado, contudo todos possuem restrições quanto a profundidade de operação. Já para os mini servomotores, o produto que mais se aproxima do grau de proteção necessário, são os mini servomotores à prova d'água fabricados pela empresa Hitec, os quais possuem grau de proteção 7 contra água com restrição de profundidade de até 1 metro.

Infelizmente tal tecnologia não satisfaria a os requisitos da missão estabelecida e muito menos a operação em maiores profundidades. Com isto, a única forma de utilizar mini servomotores neste tipo de aplicação é criando estruturas à prova d'água ou aumentar o grau de proteção do motor de alguma forma. A seguir serão mostrados alguns métodos abordadas por Frodobot (2018) em seu artigo sobre vedação de servo motores.

4.3.1 Método do Balão

Talvez um dos primeiros métodos que vem a mente quando se fala em selar um motor seja, envolver o motor com um balão de borracha. Contudo esta solução não é adequada pois pode inibir movimentos do servo e a chance do mesmo romper durante a operação é muito elevada.

4.3.2 Método do Revestimento Externo

Uma solução mais adequada é o revestimento externo do motor com substâncias abrasivas como super cola, cola quente ou ainda substâncias como a borracha sintética em formato spray ou líquida, a qual pode ser encontrada em produtos como o Plasti-Dip. Este método é um pouco melhor que o método do balão, mas também não garante a selagem do motor pois o revestimento não consegue selar o eixo do motor por ser um elemento móvel, assim permitindo micro vazamentos pelo eixo do motor. A figura 30 mostra um processo de selagem utilizando o revestimento externo com borracha sintética líquida.

4.3.3 Método da Caixa Estanque com Movimento Realizado por Hastes

Criar uma caixa estanque para o motor seja talvez a opção o método mais elegante de resolver o problema. Uma vez que o motor esta seguro contra a água dentro da caixa estanque, o empecilho ainda seria como realizar trabalho com o motor. Para

Figura 30 – Selagem de um mini servomotor utilizando o revestimento externo com borracha sintética líquida.

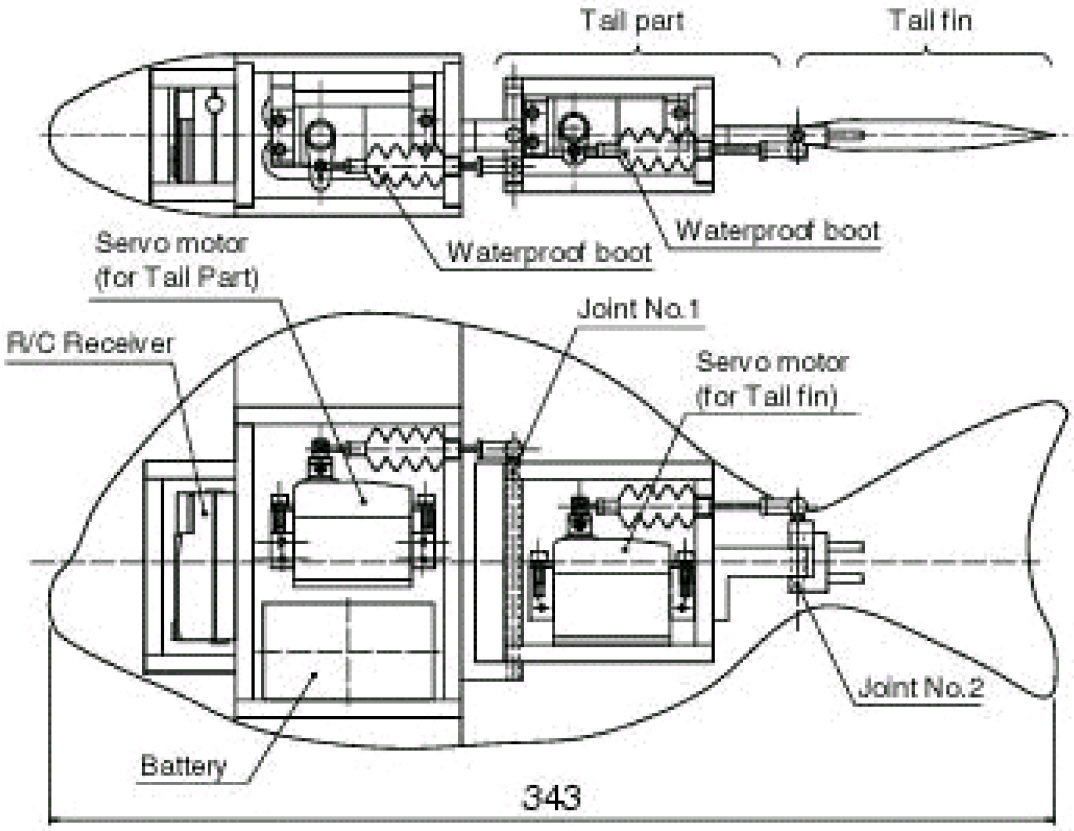
Fonte: (FRODOBOT, 2018)

isso uma haste é presa em uma das extremidades no motor enquanto a outra ponta vai para fora da caixa sendo presa a junta do manipulador. A haste atravessa a caixa através de um buraco, o qual é vedado utilizando um fole de borracha montado justo entre o servo e a parede da caixa. Este método é muito interessante , mas demanda de algum tipo de máquina de precisão ou impressora 3D para criar a caixa e também pode diminuir a manobrabilidade do manipulador, uma vez que a caixa estanque ocuparia um espaço significativo. A figura 31 mostra um robô desenvolvido para imitar um peixe, o qual utiliza o método de vedação em questão.

4.3.4 Método Combinado

Este método como o nome diz, é uma combinação de 2 métodos diferentes, o qual é considerado por Frodobot (2018) o método mais efetivo. Os métodos combinados são respectivamente o revestimento interno com mistura epoxy e o preenchimento com óleo.

Figura 31 – Robô peixe utilizando vedação do motor tipo caixa estanque com haste.



Fonte: (FRODOBOT, 2018)

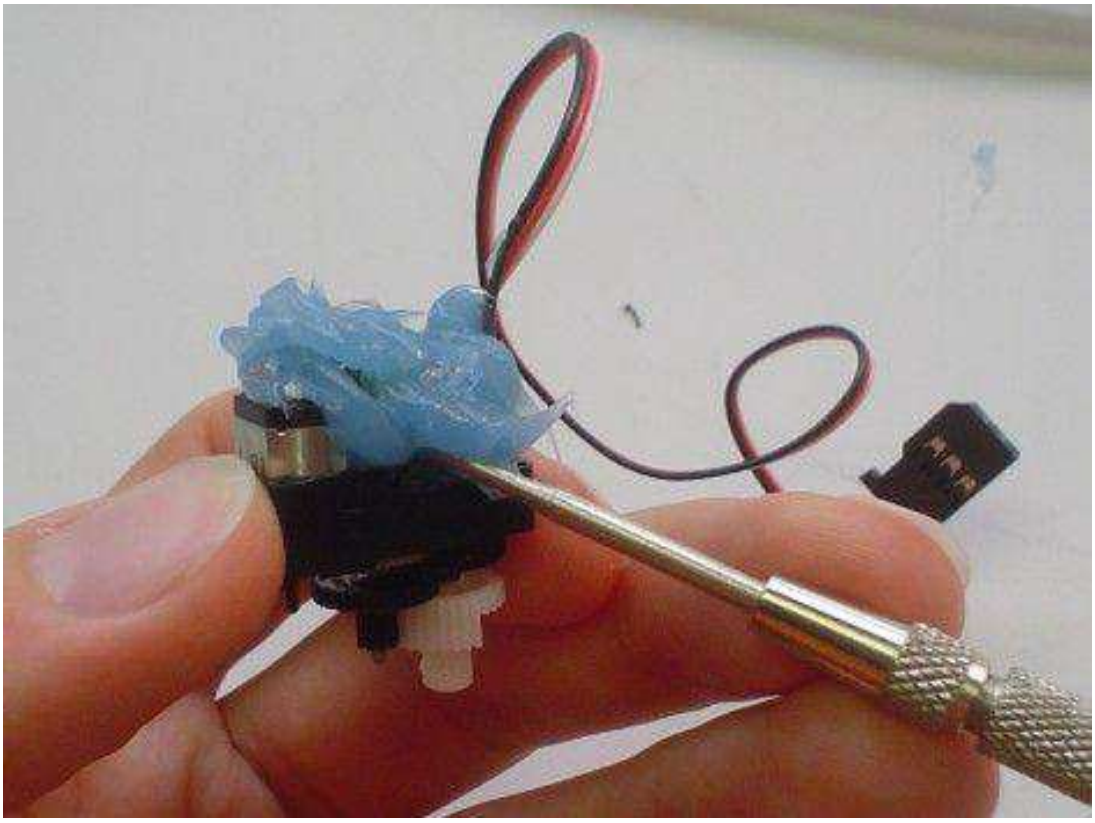
4.3.4.1 Revestimento Interno com Mistura Epoxy

O revestimento interno com resina epoxy é utilizado apenas na parte da eletrônica do mini servomotor, ao final do processo o revestimento endurece e pode travar elementos móveis. O método em questão consiste em aplicar aplicar uma mistura de resina epoxy com um catalisador não condutiva sobre a parte eletrônica do mini servomotor. Esta mistura a qual em um primeiro momento tem uma textura pastosa se molda perfeitamente ao contorno da área aplicada e depois de um tempo de cura especificado fornecedor passa para um estado sólido, desta forma garantido uma total proteção da parte eletrônica contra a água. O único problema desta solução é que ela é permanente, uma vez realizada é quase impossível reverter a condição. A figura 32 mostra uma etapa do processo de aplicação da mistura epoxy na parte eletrônica do mini servomotor.

4.3.4.2 Preenchimento com Óleo

O segundo método da combinação é o preenchimento por óleo da caixa de engrenagens do mini servomotor. Este método consiste em preencher a câmara superior do mini servomotor com óleo mineral, afim de criar uma contra pressão nas

Figura 32 – Aplicação de mistura epoxy na parte eletrônica do mini servomotor.



Fonte: (FRODOBOT, 2018)

pequenas frestas existentes. uma seringa pode ser utilizada para facilitar o trabalho. Também é necessário montar um pequeno anel de borracha entre o flange do eixo e a caixaria do mini servomotor para que o óleo na escape. A figura 33 mostra o resultado final do método combinado de vedação.

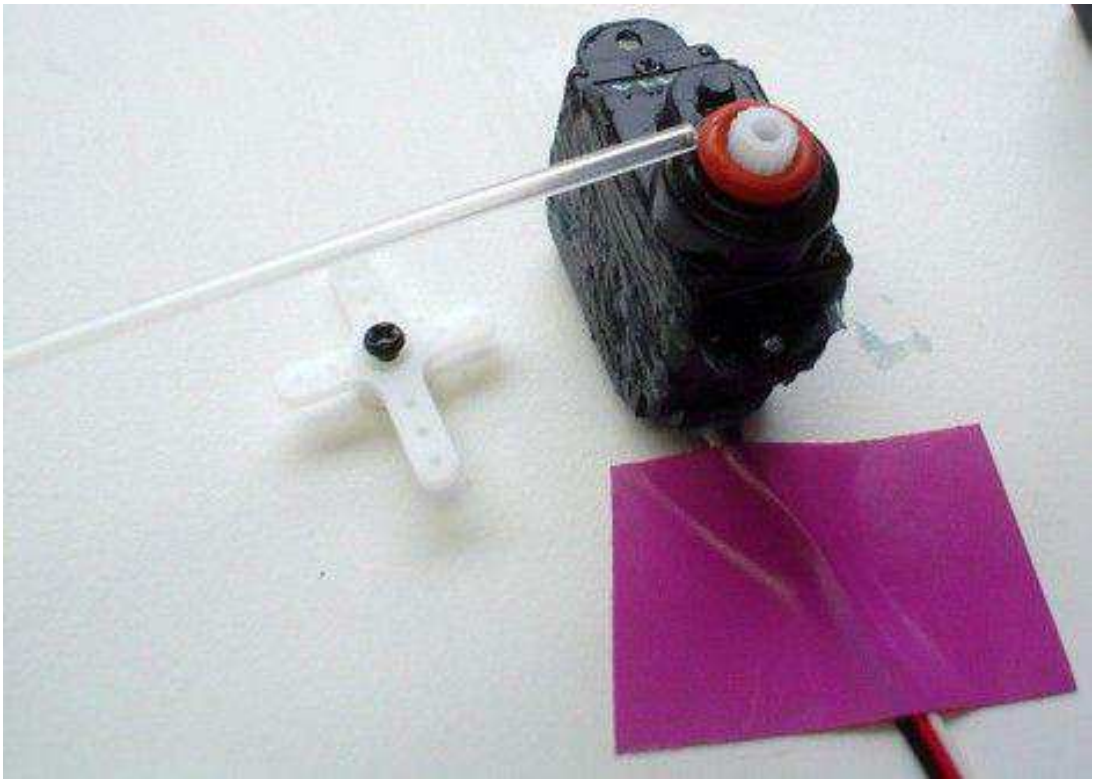
4.3.5 Outros métodos

Existem ainda outros métodos além dos citados anteriormente, contudo, grande parte deles é uma abstração dos métodos descritos. Podem ser citados como métodos alternativos os métodos de caixa estanque com sistema polia correia e caixa estanque com eixo cardan. Tais métodos utilizam a mesma estrutura do método da caixa estanque com haste mudando apenas a forma com que a junta do manipulador é atuada.

4.4 Custos

A tabela 4 descrimina os custos de cada um dos componentes do utilizados no projeto, totalizando um valor de R$ 2.221,20. Os itens com preço R$ 0,00 foram reaproveitados de outros materiais ou já haviam sido adquiridos para outros projetos.

Figura 33 – Resultado final do método combinado de vedação.



Fonte: (FRODOBOT, 2018)

Tabela 4 – Custos do projeto.

Descrição	Quantidade	Valor Unitário	Total
Conversor USB RS485	2	R$ 19,00	R$ 38,00
Servomotor Savox 1272-SG	3	R$ 430,00	R$ 1290,00
Servomotor TowerPro MG995	5	R$ 49,90	R$ 249,50
Raspberry Pi 3 Modelo B	1	R$ 299,90	R$ 299,90
Base + Suporte em U Multifuncional	3	R$ 30,00	R$ 30,00
Disco Deslizante IGUS	1	R$ 221,80	R$ 221,80
Chapa Base de Nylon	1	R$ 0,00	R$ 0,00
Eixo de 2 diâmetros da base	1	R$ 0,00	R$ 0,00
Suporte do Disco Deslizante	2	R$ 0,00	R$ 0,00
Controle XBOX 360 USB	1	R$ 0,00	R$ 0,00
Garra Robótica	1	R$ 32,00	R$ 32,00

Fonte: AUTOR (2018)

5 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho foi proposto com o intuito de desenvolver um protótipo de manipulador robótico que pudesse ser acoplado ao BlueROV2, afim de estudar quais as dificuldades e requisitos da operação de manipuladores neste ambiente.

Inicialmente foram apresentados ao leitor um conjunto de conceitos básicos sobre robótica, os quais seriam utilizados posteriormente para o entendimento do processo de escolha de componentes e tomadas de decisão de projeto. Com a problemática do BlueROV2 de não possuir meios de interagir com o ambiente apresentada, uma missão fictícia foi criada estabelecendo uma meta a ser alcançada. Conforme descrito no decorrer deste trabalho, um protótipo de manipulador robótico foi projetado, construído e testado, utilizando a comunicação serial através de um meio físico afim de tele-operar o manipulador. Contudo, todos os testes realizados com o manipulador foram feitos em bancada e portanto, com o protótipo neste estado, não foi possível comprovar os métodos de proteção contra à água descritos neste trabalho, tampouco inferir sobre a dinâmica de manipuladores no ambiente subaquático.

A evidente continuação deste trabalho seria a realização da selagem dos servo motores e a realização de experimentos no ambiente subaquático. Também seria de veras interessante, desenvolver um algoritmo de controle baseado na cinemática inversa de manipuladores seriais, tal controle diferes-se do controle junta a junta apresentado pois ao invés de controlar a posição de uma junta por vez, o parâmetro de entrada torna-se um ponto no espaço determinado pelo operador, onde o manipulador, caso seja possível, move cada uma das juntas objetivando que o efetuador final alcançe este ponto.

REFERÊNCIAS

ANTONELLI, G. **Underwater Robots – 2nd Edition**. Berlin, Heidelberg: Springer, 2006.

ANTONELLI, G.; KHATIB, O. **Springer Handbook of Robotics**. Berlin, Heidelberg: Springer, 2008.

BB SMARTWORX. **RS-422 AND RS-485 APPLICATIONS EBOOK**. Ottawa, IL, 2010. Disponível em: <http://www.bb-elec.com/Learning-Center/All-White-Papers/Serial/RS-422-and-RS-485-Applications-eBook/RS422-RS485-Application-Guide-Ebook.pdf>.

CHRIST, R. D.; WERNLI, R. L. **The ROV Manual: A User Guide for Remotely Operated Vehicles**. Kidlington, Oxford: Butterworth-Heinemann, 2013.

FRODOBOT. **Water-proofing a Servo**. 2018. Disponível em: <https://www.instructables.com/id/Water-proofing-a-Servo/>.

IGUS. **Product Description**. 2018. Disponível em: <https://www.igus.com.br/info/plain-bearings-iglidur-prt-slewing-bearing-type-02>.

ISO. 8373: 2012. robots and robotic devices–vocabulary. **International Standardization Organization (ISO)**, 2012.

NOAA, U. **How much of the ocean have we explored?** 2018. Disponível em: <https://oceanservice.noaa.gov/facts/exploration.html>.

OPUS. **Índice de proteção (IP)**. 2015. Disponível em: <http://opusled.com.br/blog/index.php/indice-de-protecao-para-iluminacao/>. Acesso em: dez. 2018.

PI, F. R. **Product Description**. 2018. Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>.

ROBERTS, A. The history of science fiction. 01 2016.

ROBOTICS, B. **Product Description**. 2018. Disponível em: <https://www.bluerobotics.com/store/rov/bluerov2/>.

SALVI, A. Z. Avaliação de estratégias de comunicação para um veículo subaquático de operação remota. **Relatório de Estágio**, 2018.

SAVOX. **Product Description**. 2018. Disponível em: <https://www.savoxusa.com/products/savsv1272sg-hv-coreless-digital-servo/>.

SICILIANO, B.; SCIAVICCO, L. **Robotics - Modelling, Planning and Control**. Berlin, Heidelberg: Springer, 2010.

TOWERPRO. **Product Description**. 2018. Disponível em: <https://www.towerpro.com.tw/product/mg995/>.

TSAI, L.-W. **Robot Analysis: The Mechanics of Serial and Parallel Manipulators**. [S.l.]: John Wiley & Sons, 1999.

USN, U. S. N. **Cable-Controlelled Underwater Recovery Vehicle (CURV)**. 2016. Disponível em: <http://www.public.navy.mil/>.

APÊNDICE A - APLICAÇÃO DO COMPUTADOR

```
1  import inputs
2  import serial
3  import struct
4
5  ser = serial.Serial()
6  ser.baudrate = 9600
7  ser.port = 'COM6'
8  ser.bytesize = serial.EIGHTBITS
9  ser.parity = serial.PARITY_NONE
10 ser.stopbits = serial.STOPBITS_ONE
11 ser.timeout = None
12
13 ser.open()
14
15 a=b=c=d=e=r=s= 0
16
17 while(1):
18
19     msg = struct.pack('BBBBBBB',r,a,b,c,d,e,s)
20
21     ser.write(msg)
22
23     events = inputs.get_gamepad()
24
25     print(events)
26     for event in events:
27         print (event.ev_type, event.code, event.state)
28         if(event.code == 'BTN_EAST' or event.code == 'BTN_WEST' or
29            event.code == 'BTN_NORTH' or event.code == 'BTN_SOUTH' or
30            event.code == 'BTN_SELECT' or event.code == 'BTN_START' or
31            event.code == 'ABS_HAT0Y' or event.code == 'ABS_HAT0X' or
32            event.code == 'BTN_TR' or event.code == 'BTN_TL'):
33
34
35             #JUNTA 01
36             if (event.code == 'BTN_EAST' and event.state == 1):
37                 a = 1
38             elif (event.code == 'BTN_WEST' and event.state == 1):
39                 a = 2
```

```
40              elif ((event.code == 'BTN_WEST' and event.state == 0) or
41                    (event.code == 'BTN_EAST' and event.state == 0)):
42                  a = 0
43              else:
44                  a = a
45
46              #JUNTA 02
47              if (event.code == 'BTN_NORTH' and event.state == 1):
48                  b = 1
49              elif (event.code == 'BTN_SOUTH' and event.state == 1):
50                  b = 2
51              elif ((event.code == 'BTN_SOUTH' and event.state == 0)or
52                    (event.code == 'BTN_NORTH' and event.state == 0)):
53                  b = 0
54              else:
55                  b = b
56
57              #BACK
58              if (event.code == 'BTN_SELECT' and event.state == 1 and r == 0):
59                  r = 1
60              elif (event.code == 'BTN_SELECT' and event.state == 1 and r == 1):
61                  r = 0
62
63              #START
64              if (event.code == 'BTN_START' and event.state == 1 and s == 0):
65                 s = 1
66              elif (event.code == 'BTN_START' and event.state == 1 and s ==1):
67                 s = 0
68
69
70              #JUNTA 03
71              if (event.code == 'ABS_HAT0Y' and event.state ==  1):
72                  c = 1
73              elif (event.code == 'ABS_HAT0Y' and event.state == 1):
74                  c = 2
75              elif (event.code == 'ABS_HAT0Y' and event.state == 0):
76                  c = 0
77              else:
78                  c = c
79
80              #JUNTA 04
81              if (event.code == 'ABS_HAT0X' and event.state == 1):
82                  d = 1
83              elif (event.code == 'ABS_HAT0X' and event.state ==  1):
84                  d = 2
85              elif (event.code == 'ABS_HAT0X' and event.state == 0):
86                  d = 0
```

```
87              else:
88                  d = d
89
90              #GARRA
91              if (event.code == 'BTN_TR' and event.state == 1):
92                  e = 1
93              elif (event.code == 'BTN_TL' and event.state == 1):
94                  e = 2
95              elif ((event.code == 'BTN_TR' and event.state == 0)or
96                    (event.code == 'BTN_TL' and event.state == 0)):
97                  e = 0
98              else:
99                  e = e
```

APÊNDICE B - APLICAÇÃO DO RASPBERRY PI

```
1  import serial
2  import struct
3  import pigpio
4  from time import sleep
5
6  def  angle_Up(GpIO,LH):
7      if(pi.get_servo_pulsewidth(GpIO) < LH):
8          pi.set_servo_pulsewidth(GpIO,pi.get_servo_pulsewidth(GpIO)+5)
9
10 def angle_Down(GpIO,LD):
11     if(pi.get_servo_pulsewidth(GpIO) > LD):
12         pi.set_servo_pulsewidth(GpIO,pi.get_servo_pulsewidth(GpIO) 5)
13
14
15 ser = serial.Serial()
16 ser.baudrate = 9600
17 ser.port = '/dev/ttyUSB0'
18 ser.bytesize = serial.EIGHTBITS
19 ser.parity = serial.PARITY_NONE
20 ser.stopbits = serial.STOPBITS_ONE
21 ser.timeout = None
22
23 ser.open()
24
25 GPIO.setwarnings(False)
26
27 pi= pigpio.pi()
28
29 pi.set_mode(17,pigpio.OUTPUT)
30 pi.set_PWM_frequency(17,50)
31 pi.set_servo_pulsewidth(17,1500)
32
33 pi.set_mode(18,pigpio.OUTPUT)
34 pi.set_PWM_frequency(18,50)
35 pi.set_servo_pulsewidth(18,1750)
36
37 pi.set_mode(27,pigpio.OUTPUT)
38 pi.set_PWM_frequency(27,50)
39 pi.set_servo_pulsewidth(27,1500)
```

```
40
41 pi.set_mode(22,pigpio.OUTPUT)
42 pi.set_PWM_frequency(22,50)
43 pi.set_servo_pulsewidth(22,1500)
44
45 pi.set_mode(23,pigpio.OUTPUT)
46 pi.set_PWM_frequency(23,50)
47 pi.set_servo_pulsewidth(23,750)
48
49
50 while(1):
51
52     msg = struct.unpack('BBBBBBB',ser.read(7))
53
54     while(msg[0]==1):
55
56         msg = struct.unpack('BBBBBBB',ser.read(7))
57
58         if(msg[6]==1)
59
60             #JUNTA 01
61             if (msg[1]==1):
62                 angle_Up(17,2000)
63             elif (msg[1]==2):
64                 angle_Down(17,1000)
65
66             #JUNTA 02
67             if (msg[2]==1):
68                 angle_Up(18,2000)
69             elif (msg[2]==2):
70                 angle_Down(18,1500)
71
72             #JUNTA 03
73             if (msg[3]==1):
74                 angle_Up(27,2500)
75             elif (msg[3]==2):
76                 angle_Down(27,500)
77
78             #JUNTA 04
79             if (msg[4]==1):
80                 angle_Up(22,2500)
81             elif (msg[4]==2):
82                 angle_Down(22,500)
83
84             #GARRA
85             if (msg[5]==1):
86                 angle_Up(23,1000)
```

```
87                elif (msg[5]==2):
88                    angle_Down(23,500)
89
90                sleep(0.01)
91
92        pi.set_servo_pulsewidth(17,0)
93        pi.set_servo_pulsewidth(18,0)
94        pi.set_servo_pulsewidth(27,0)
95        pi.set_servo_pulsewidth(22,0)
96        pi.set_servo_pulsewidth(23,0)
```