UNIVERSIDADE DE SÃO PAULO

ESCOLA DE ENGENHARIA DE SÃO CARLOS

DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DE COMPUTAÇÃO

SISTEMA DE MAPEAMENTO TRIDIMENSIONAL DE AMBIENTES

Guilherme Torres Ferreira

Orientador: Prof. Dr. Valdir Grassi Jr.

São Carlos

2014

GUILHERME TORRES FERREIRA

SISTEMA DE MAPEAMENTO TRIDIMENSIONAL DE AMBIENTES

Trabalho de Conclusão de Curso apresentado
à Escola de Engenharia de São Carlos, da
Universidade de São Paulo

Curso de Engenharia Elétrica com ênfase em
Eletrônica

ORIENTADOR: Prof. Dr. Valdir Grassi Jr.

São Carlos
2014

Ferreira, Guilherme Torres
F383s Sistema de Mapeamento Tridimensional de Ambientes / Guilherme Torres Ferreira; orientador Valdir Grassi Jr. São Carlos, 2014.

Monografia (Graduação em Engenharia Elétrica com ênfase em Eletrônica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2014.

1. Mapeamento tridimensional. 2. LIDAR. 3. Robótica. 4. Point Cloud. I. Título.

FOLHA DE APROVAÇÃO

Nome: Guilherme Torres Ferreira

Titulo: "Sistema de mapeamento tridimensional de ambientes"

Trabalho de Conclusão de Curso defendido e aprovado em 24 / 11 / 2014,

com NOTA 9,0 (nove , zero), pela Comissão Julgadora:

Prof. Dr. Valdir Grassi Júnior - (Orientador - SEL/EESC/USP)

Mestre Mauricio Eiji Nakai - (Doutorando - SEL/EESC/USP)

Prof. Dr. Fernando Santos Osório - (SSC/ICMC/USP)

Coordenador da CoC-Engenharia Elétrica - EESC/USP:
Prof. Associado Homero Schiabel

Dedico este trabalho à minha querida mãe Ada (in memoriam) e ao meu querido pai Adaías; graças ao grande esforço deles em minha educação, esta realização se tornou possível

Agradecimentos

Agradeço primeiramente a Deus pelo dom da vida, proteção e oportunidades que Ele me concede. Agradeço aos meus pais e meu irmão que sempre me apoiaram e me encorajaram ao longo da trajetória na universidade.

Agradeço à USP e aos professores da EESC pelo excelente ambiente de aprendizado que proporcionaram durante o curso. Agradeço ao meu orientador Prof. Valdir Grassi Jr. pelo apoio e compreensão ao decorrer deste trabalho.

Agradeço aos meus amigos de São Carlos que tornaram os anos de universidade muito especiais. A todos, muito obrigado!

Sumário

Lista de Figuras

Resumo

A área de mapeamento 3D está em constante expansão devido à alta demanda por dados tridimensionais em aplicações como robótica, arquitetura, engenharia civil, etc. Este trabalho apresenta o desenvolvimento e avaliação de um sistema de mapeamento tridimensional. Dentre as tecnologias possíveis para mapeamento tridimensional, é utilizado neste trabalho o método baseado em LIDAR. O sistema é implementado com um sensor LIDAR 2D que, acoplado ao eixo de um motor elétrico, gera mapas tridimensionais do ambiente analisado. Dadas as características do sensor utilizado, o sistema aplica-se a ambientes internos e detecta objetos e superfícies a um raio máximo de 4 metros. A partir das distâncias medidas pelo sensor, as coordenadas euclidianas de pontos na superfície são definidas *utilizando* matrizes de transformação homogênea. Os pontos obtidos são então representados em uma *Point Cloud*, permitindo ao usuário uma interação prática e intuitiva com o modelo gerado. O desempenho do sistema foi testado em diferentes ambientes, comparando resolução, definição e tempo de mapeamento.

Palavras-chave: Mapeamento tridimensional, LIDAR, Robótica, *Point Cloud*

Abstract

The field of 3D mapping is in constant expansion due to the high demand for three-dimensional data in applications such as robotics, architecture, civil engineering, etc. This text presents the development and evaluation of a three-dimensional mapping system. Among the possible technologies used for three-dimensional mapping, the one applied in this project is based on LIDAR. The system is implemented with a 2D LIDAR sensor coupled to the shaft of an electric motor. As the shaft rotates, the sensor is able to generate three-dimensional maps of the environment. Due to the sensor characteristics, the system applies only to indoor environments and detects surfaces and objects within 4 meters. From the distances measured by the sensor, the coordinates of points on the surface are calculated using homogeneous transformation matrices. The points are then represented in a Point Cloud, allowing the user a convenient and intuitive interaction with the generated map. The system performance was tested in different environments, comparing resolution, definition and mapping time.

Keywords: 3D mapping, LIDAR, Robotics, Point Cloud

1. Introdução

A utilização de modelos tridimensionais de objetos e ambientes tem se tornado cada vez mais comum, principalmente nas duas últimas décadas, graças aos recursos de computação cada vez mais avançados e ao surgimento e aprimoramento tecnológico de dispositivos de mapeamento tridimensional (*scanners*). Muitas áreas se beneficiam da utilização de modelos tridimensionais, alguns exemplos são: na indústria, para o projeto e manutenção de plantas industriais; na medicina, para representação e estudo de anatomia; em arquitetura, para projetos de interiores.

Modelos tridimensionais podem ser gerados à mão com auxílio de ferramentas CAD; através de algoritmos de modelagem procedural; ou utilizando dispositivos de mapeamento. No último caso citado, diferentes tecnologias são utilizadas para geração de modelos. As técnicas consistem em definir as coordenadas tridimensionais de pontos na superfície dos objetos analisados e, a partir da combinação e representação gráfica destes pontos, gerar um modelo do ambiente real.

1.1. Motivação

A geração de mapas tridimensionais de ambientes pode ser útil em uma gama de aplicações como engenharia civil, arquitetura, entretenimento, projetos de plantas industriais e processos de engenharia reversa.

Um dos propósitos de se desenvolver um sistema de mapeamento tridimensional é a sua aplicação em robótica. O LASI (Laboratório de Sistemas Inteligentes, na USP São Carlos) desenvolve projetos científicos na área de robótica, automação e controle de sistemas. Dentre os trabalhos realizados neste laboratório, podemos citar o controle de robôs autônomos e a instrumentação dos mesmos. Um dos robôs para o qual é desenvolvida instrumentação no LASI é o Pioneer 3-AT. Um robô autônomo pode se utilizar do mapa para se localizar e navegar no ambiente, como também pode ser utilizado para a geração do mapa 3D. Portanto, a motivação inicial do trabalho surgiu da possibilidade de integrar um sistema de mapeamento 3D ao Pioneer 3-AT e outros robôs do LASI.

1.2. Objetivos

Este trabalho tem como objetivo o desenvolvimento de hardware e software de um sistema de mapeamento tridimensional de ambientes. A parte de hardware envolve a escolha de componentes e projeto de peças para o sistema, enquanto a parte de software tem como escopo a definição de ferramentas computacionais a serem utilizadas, integração dos dispositivos e elaboração do código. O sistema é aplicável em ambientes interiores e deve prover ao usuário uma visualização do mapa gerado.

1.3. Organização do Trabalho

A fim de proporcionar uma visão geral da estrutura do trabalho, o conteúdo resumido dos capítulos é apresentado a seguir. O trabalho é dividido em cinco capítulos:

O **capítulo 1** apresenta uma contextualização ao tema de mapeamento tridimensional, trata da motivação e objetivo do trabalho, e por fim, apresenta a estrutura de organização do trabalho.

O **capítulo 2** apresenta uma revisão dos trabalhos relacionados ao tema de mapeamento tridimensional, citando os métodos existentes e exemplos de aplicação em trabalhos científicos.

O **capítulo 3** descreve o projeto de hardware e software do sistema de mapeamento tridimensional, incluindo a fundamentação teórica utilizada para sua elaboração.

No **capítulo 4**, são discutidos os resultados da aplicação do sistema projetado, apresentando mapas gerados pelo sistema e analisando seu desempenho.

Por fim, o **capítulo 5** traz as conclusões do trabalho, além de considerações acerca de trabalhos futuros.

2. Revisão Bibliográfica

2.1. Mapeamento Tridimensional de Ambientes

O avanço de sistemas computacionais tem tornado possível e cada vez mais constante a utilização de modelos tridimensionais para análises e projetos, que permitem uma interação intuitiva e mais próxima da realidade entre o observador e objeto em estudo. Um destes recursos trata-se do mapeamento 3D de ambientes, cujas aplicações são bem diversificadas. Um exemplo de uso recorrente desta tecnologia é na Arquitetura. Laing e Scott (2011) utilizaram um sistema de mapeamento 3D para registrar interiores de edifícios históricos de centros urbanos, o que proporcionou, além de uma maneira mais eficiente e precisa de armazenar dados sobre o legado arquitetônico, também um ganho na elaboração de estratégias de conservação dessas construções. O mapeamento de ambientes também pode ser usado em cultura e entretenimento, como na geração de modelos de museus e universidades para realização de visitas virtuais e modelos para jogos de vídeo game e filmes (TURNER et al., 2014). Outro exemplo de aplicação é a exploração de ambientes hostis ou de difícil acesso, como utilizado por Hähnel e Montemerlo (2002) que desenvolveram um sistema de mapeamento volumétrico para minas subterrâneas baseado em um robô autônomo equipado de diversos sensores de varredura laser. Por sinal, as aplicações em robótica compõem a grande maioria de produções científicas sobre este tema.

A robótica é um campo de pesquisa extremamente importante para o tema de mapeamento 3D de ambientes por se tratar de uma área cujos conhecimentos são utilizados tanto para a geração de mapas, como para a utilização dos mesmos em diversas aplicações como navegação, detecção de obstáculos, exploração de ambientes e automação industrial.

2.1.1. Métodos de Mapeamento Tridimensional

Existem diversas tecnologias utilizadas para mapeamento tridimensional, no entanto a literatura destaca três métodos principais: triangulação, câmeras do tipo Visão Estéreo, além de sensores de varredura a laser, mais conhecido como LIDAR. (MORALES et al., 2011)(WULF; WAGNER, 2003) A seguir, os aspectos fundamentais de funcionamento destes métodos são

descritos, bem como as aplicações mais apropriadas para cada um deles e grupos de pesquisa que os utilizaram.

2.1.1.1. Triangulação

O método de mapeamento 3D por triangulação consiste em um emissor laser que projeta padrões geométricos, em geral um ponto ou uma reta, sobre o objeto ou superfície que se deseja escanear. Além disso, uma câmera digital é posicionada a uma distância d do emissor de laser, formando assim um triângulo entre o emissor, a projeção sobre a superfície e a própria câmera. O sensor da câmera captura a posição do padrão geométrico projetado na superfície. Como a distância d entre o emissor e câmera, o ângulo θ de rotação do emissor e as propriedades focais da câmera são conhecidos, é possível determinar a coordenada (x, y, z) de um determinado ponto na superfície analisada através de relações geométricas e trigonométricas (FRANÇA; GAZZIRO, 2005). A Figura 2.1 apresenta a configuração básica do método de mapeamento 3D por triangulação.

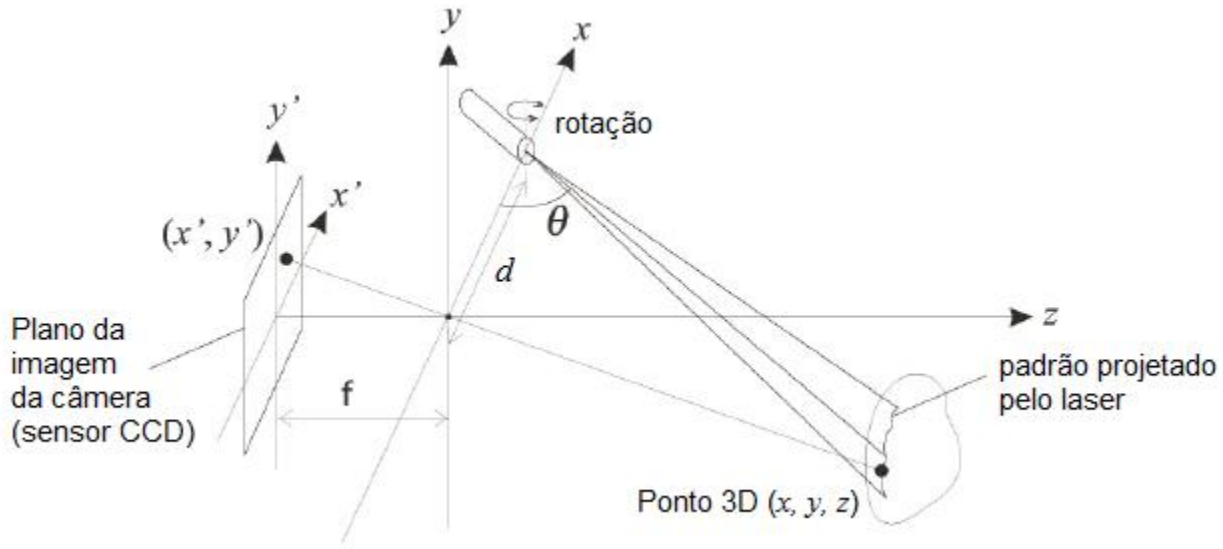


Figura 2.1 - Esquema de funcionamento do método de triangulação
Fonte: Adaptado de França e Gazziro (2005)

Na Figura 2.1, o ponto (x', y') está contido no plano da imagem registrada pela câmera e representa a coordenada do pixel que capturou o ponto projetado pelo laser na superfície. O plano da imagem está a uma distância focal f da lente da câmera. Portanto, conhecendo as

propriedades focais e do sensor CCD[1] da câmera, que determinam a posição do pixel na imagem, e as relações geométricas entre emissor e câmera supracitadas, as coordenadas (x, y, z) de um ponto na superfície podem ser calculadas utilizando as seguintes equações:

$$x = \frac{dx'}{f \operatorname{cotg} \theta - x'} \tag{2.1}$$

$$y = \frac{dy'}{f \operatorname{cotg} \theta - x'} \tag{2.2}$$

$$z = \frac{df}{f \operatorname{cotg} \theta - x'} \tag{2.3}$$

Quando comparado a outras tecnologias, o método de triangulação apresenta altíssima resolução, no entanto o alcance é apenas da ordem de alguns metros o que inviabiliza o mapeamento de grandes ambientes ou objetos que estejam fora do campo de visão da câmera ou do alcance do laser. Sendo assim, este método é mais apropriado para mapeamento de objetos pequenos e processos de engenharia reversa. (MORALES et al., 2011).

Também pode-se considerar como uma vantagem significativa do método de triangulação o seu baixo custo de implementação. Aydar, Akyol, e Duran (2011) desenvolveram um sistema de baixo custo baseado em triangulação que foi utilizado com excelentes resultados para escanear objetos pequenos, porém complexos, como esculturas.

2.1.1.2. Visão Estéreo

Outro importante método de obtenção de dados tridimensionais sobre objetos e superfície é o de visão estéreo. O funcionamento desta técnica baseia-se no sistema de visão de seres humanos e animais; duas câmeras são posicionadas a uma pequena distância uma da outra, de forma que as duas apontem para a mesma região do espaço que se deseja mapear. A partir da combinação das imagens 2D obtidas pelas câmeras é possível obter dados de profundidade espacial do objeto em estudo, assim como são formadas imagens tridimensionais pelo sistema de visão humano. A combinação de imagens deve ser realizada no intuito de identificar pontos equivalentes nas imagens das duas câmeras, no que é denominado *problema de correspondência*. Tendo identificado um ponto equivalente sob as duas perspectivas, relações geométricas semelhantes às utilizadas no método de triangulação

[1] Sensor CCD (Charge-coupled Device) é dispositivo semicondutor, composto por células fotoelétricas, utilizado para captação de imagens

são suficientes para calcular a dimensão de profundidade da superfície (CHEN et al., 2008). A Figura 2.2 apresenta um esquema simplificado do método de visão estéreo. Neste esquema, os pontos $u_L(x_L, y_L)$ e $u_R(x_R, y_R)$ representam as projeções do ponto *P* do mundo real em imagens obtidas pelas câmeras esquerda e direita, respectivamente. O parâmetro de disparidade é definido como a distância entre as projeções u_L e u_R. A distância *b* entre as duas câmeras é denominada linha de base. Portanto, é possível calcular as coordenadas (x, y, z) do ponto *P* através das seguintes expressões:

$$x = x_L \frac{f}{z} \quad \text{ou} \quad x = x_R \frac{f}{z} + b \tag{2.4}$$

$$y = y_L \frac{z}{f} \quad \text{ou} \quad y = y_R \frac{z}{f} \tag{2.5}$$

$$z = f \frac{b}{(x_L - x_R)} \tag{2.6}$$

O método de visão estéreo é especialmente interessante para aplicações de exploração espacial por ser uma tecnologia passiva, portanto menos energia é necessária para seu funcionamento, podendo ser alimentado por células solares. Este foi o conceito utilizado por Goldberg (2002), que descreve o projeto de um robô explorador para a superfície de Marte equipado com um sistema de visão estéreo para navegação, desvio de obstáculos e registro de objetos encontrados na superfície.

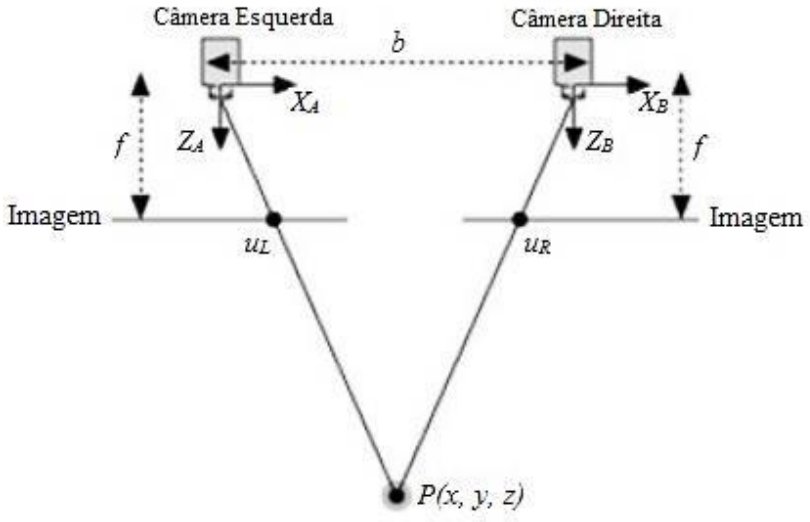


Figura 2.2 - Esquema de funcionamento do método de Visão Estéreo
Fonte: Adaptado de National Instruments (2012) – http://www.ni.com/white-paper/14103/pt/

2.1.1.3. LIDAR

O sistema de varredura a laser, popularmente conhecido como LIDAR (da sigla em inglês *Light Detection And Ranging*), é um dos métodos mais utilizados para mapeamento tridimensional. Basicamente, um dispositivo LIDAR é composto por um emissor de luz e um fotorreceptor. A técnica se baseia em determinar as distâncias entre o dispositivo e pontos da superfície a ser mapeada através da emissão de uma onda de luz e detecção da onda refletida pela superfície. Ao incidir sobre a superfície, parte da luz emitida é refletida na direção do dispositivo, que identifica a onda através do fotorreceptor. (WUTKE, 2006)

Existem primariamente duas maneiras de calcular a distância através deste método. A primeira delas é denominada *time of flight,* na qual se mede o tempo entre a emissão de um pulso de luz e a recepção do pulso refletido. A distância entre o dispositivo LIDAR e a superfície pode ser então calculada utilizando a seguinte expressão:

$$d = \frac{1}{2}ct \tag{2.7}$$

na qual t é o tempo medido entre a emissão e recepção do pulso e $c \approx 299.792.458\, m/s$ é a velocidade da luz. A Figura 2.3 apresenta um esquema de funcionamento do método *time of flight*

A segunda estratégia de cálculo da distância tem como princípio a diferença de fase entre uma onda emitida e a onda refletida pela superfície. Neste caso, a onda de luz a ser emitida é modulada senoidalmente e incidida sobre a superfície. A componente da onda refletida em direção ao sensor é então detectada e comparada com a onda emitida para determinação da diferença de fase entre as duas. Dado que as ondas emitida e refletida possuem mesma frequência f, é possível determinar a distância entre o dispositivo de mapeamento e a superfície a partir da seguinte expressão:

$$d = \frac{c\varphi}{4\pi f} \tag{2.8}$$

na qual φ é a diferença de fase entre as ondas emitida e recebida em radianos (KEMENY; TURNER, 2008). A Figura 2.4 apresenta um esquema do método de diferença de fase.

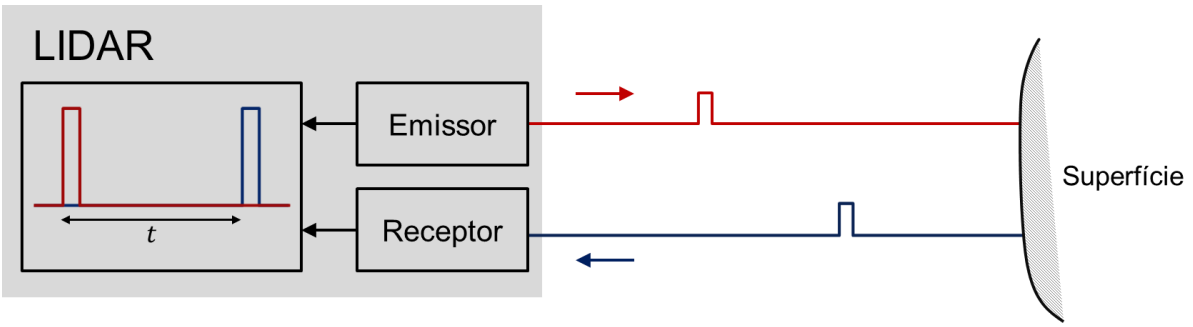


Figura 2.3 - Esquema de funcionamento do LIDAR - Time of Flight
Fonte: Autoria própria

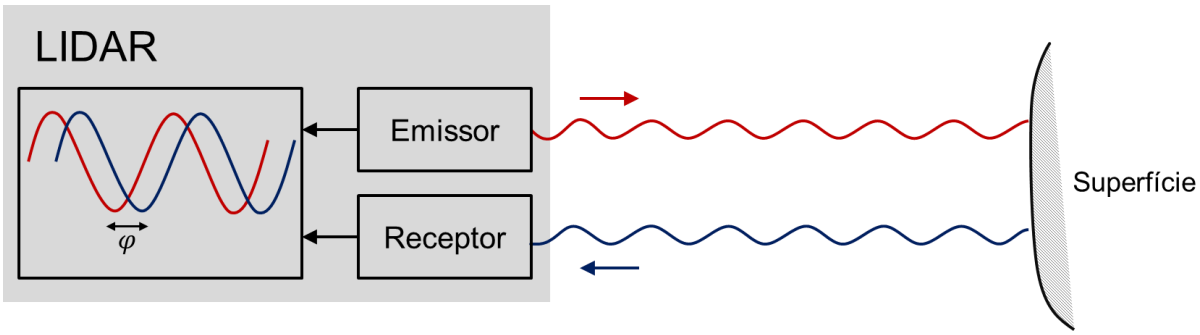


Figura 2.4 - Esquema de funcionamento do LIDAR - Diferença de fase
Fonte: Autoria própria

A varredura de pontos em um plano da superfície pode ser realizada com auxílio de dispositivos optomecânicos. Cameron, Szumski e West (1991) descreveram um sistema de varredura laser composto de um transmissor de laser modulado, um espelho poligonal rotacionado por um servomotor, um fotorreceptor e um módulo de processamento. À medida que o servomotor é acionado, o espelho reflete a luz emitida pelo transmissor em uma direção específica, realizando uma varredura em um plano da superfície. O aspecto poligonal do espelho permite que as ondas refletidas pela superfície sejam captadas pelo fotorreceptor que envia os sinais a um processador que calcula as respectivas distâncias a cada ponto da superfície varrida. Existem vários dispositivos comerciais que seguem este princípio de construção e são muito comuns em aplicações de robótica bidimensional como desvios de obstáculos. Alguns sistemas LIDAR utilizam elementos ópticos em mais de um eixo para fazer aquisição de dados 3D de superfícies. No entanto, esse tipo de dispositivo, em geral utilizado em topografia, não atende aos requisitos de aplicações em robótica e automação industrial por ser muito pesado, apresentar um custo elevado e por ser consideravelmente lento (o mapeamento pode levar alguns minutos). Uma alternativa que tem sido bastante utilizada é a

conversão de sistemas de varredura laser 2D em 3D através da adição de um eixo de rotação mecânico.

A combinação de um dispositivo LIDAR 2D e um motor elétrico, que possibilita a aquisição de dados tridimensionais de ambientes por meio da adição de um grau de liberdade, é interessante por conta do baixo custo de implementação e da versatilidade. Existem três métodos possíveis para realização desta montagem, cada um referente à rotação do plano de varredura do laser em torno de um eixo de um sistema de coordenadas fixo. Estes métodos são denominados *pitching, yawing* e *rolling.* Wulf e Wagner (2003) fazem uma análise detalhada destes métodos, comparando o desempenho de cada um em cenários específicos, inclusive em aplicações dinâmicas que demandam rapidez no escaneamento. Foram explicitados os pontos fortes e fracos de cada configuração de forma a otimizar a escolha do método mais apropriado para uma determinada aplicação. A seguir, os aspectos principais destas configurações são apresentados.

2.1.1.3.1. Rolling

Na configuração *rolling* (Figura 2.5), o dispositivo LIDAR é rotacionado em torno do eixo horizontal cuja direção é coincidente à reta que divide pela metade o campo de varredura do laser. Para um mapeamento completo da região frontal, é necessário rotacionar o sensor em 180°. Como a densidade de pontos coletados é maior à frente do sensor onde se localiza o ponto focal, essa configuração se torna apropriada para aplicações que demandam um nível de detalhe mais preciso nessa região, como por exemplo, detecção e desvio obstáculos. (TÂCHE et al., 2011)

Uma grande vantagem deste método é que, devido ao sentido de rotação do sensor ser o mesmo do motor, o acoplamento mecânico entre estes elementos é relativamente mais simples, visto que não são necessários elementos mecânicos para conversão do sentido de rotação. Isso permite que a montagem seja mais leve e ocupe menos espaço, o que é uma característica desejável para aplicações em robôs. A configuração também permite que o centro de massa do sensor esteja próximo ao eixo de rotação do motor, o que alivia o torque requisitado ao motor e, portanto diminui custos de implementação do sistema (WATSON; ULRICH, 2014).

Como a maioria dos sensores possui uma área não coberta pela varredura do laser, isso configura um ponto negativo na aplicação de *rolling*, pois apenas a região à frente do sensor será mapeada.

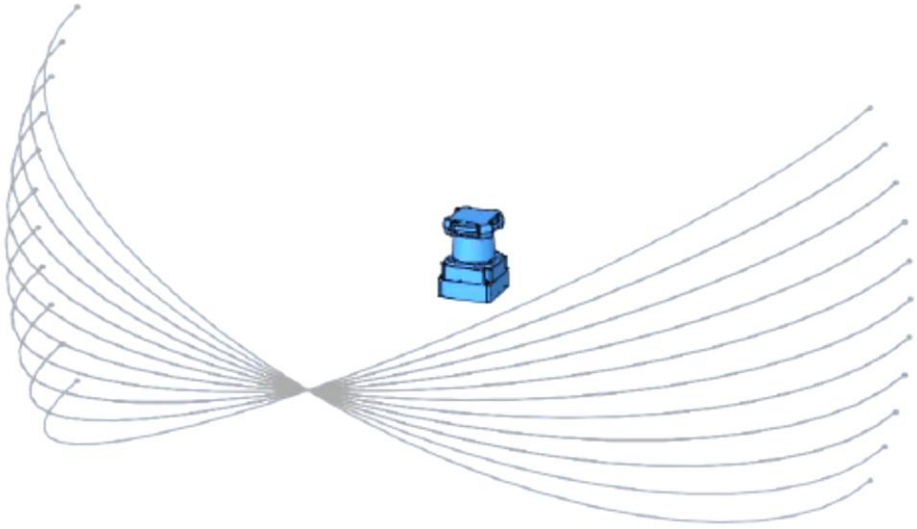


Figura 2.5 - Aspecto de cobertura na configuração *rolling*
Fonte: Watson e Ulrich (2014)

2.1.1.3.2. Pitching

Nesta configuração, o sensor é rotacionado em torno do eixo horizontal e perpendicular ao eixo de rotação de *rolling* (Figura 2.6). As regiões com maior densidade de pontos medidos estão à esquerda e direita do sensor, enquanto que a região frontal apresenta um mapeamento mais esparso e uniforme. Informações sobre a região frontal ao sensor são obtidas com mais rapidez do que o método de *rolling*, o que sugere a utilização da configuração de pitching em aplicações dinâmicas, que demandem tomadas de decisão ágeis, como em navegação de robôs móveis e monitoramento de áreas. (MORALES et al., 2011).

Em contrapartida, o aparato mecânico necessário para o acoplamento entre sensor e motor é mais complexo, necessita de mais elementos mecânicos e ocupa mais espaço, dado que o acoplamento direto ao eixo do motor não é viável, pois o mesmo obstruiria parte do campo de visão do sensor (WATSON; ULRICH, 2014).

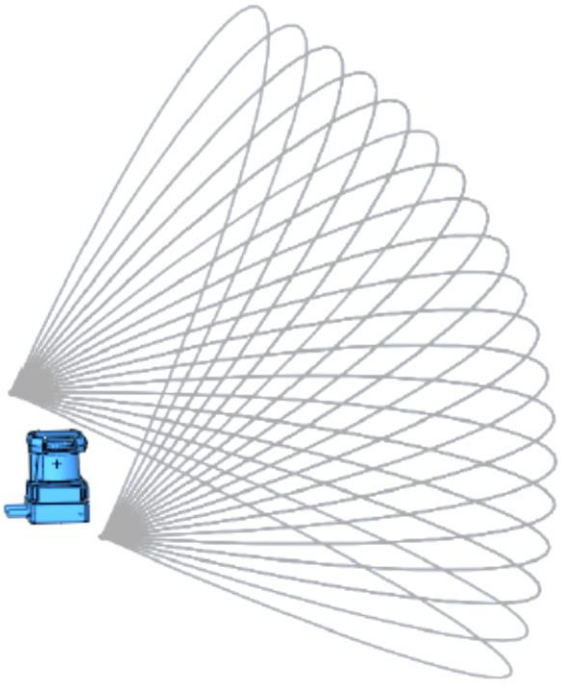


Figura 2.6 - Aspecto de cobertura na configuração *pitching*
Fonte: Watson e Ulrich (2014)

2.1.1.3.3. Yawing

O método de *yawing* baseia-se na rotação do sensor em torno de um eixo vertical (Figura 2.7). Para que a rotação do sensor não seja redundante ao plano de varredura do laser, duas configurações alternativas podem ser usadas; posicionando o sensor com uma de suas laterais ou com a parte de trás do sensor voltada para baixo. Esse é o método que permite um mapeamento em 360º de forma mais uniforme, pois as regiões com maior densidade de pontos medidos estão localizadas em áreas, na maioria dos casos, de pouco interesse, acima e abaixo do sensor. Portanto, essa configuração é a mais apropriada, quando é necessária uma visão do ambiente superior à 180°, pois nessa condição o *yawing* obtém resultados melhores que *rolling* e *pitching* (WULF; WAGNER, 2003).

O acoplamento mecânico depende da configuração adotada; caso seja a lateral (com uma das laterais voltadas para baixo), a implementação será semelhante à utilizada no método de *pitching.* No outro caso (com a parte de trás do sensor voltada para baixo), a montagem se assemelhará a de *rolling*.

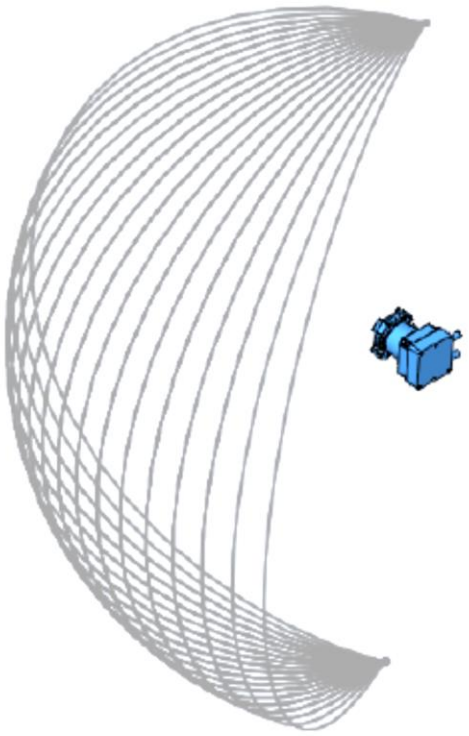


Figura 2.7 - Aspecto de cobertura na configuração *yawing*
Fonte: Watson e Ulrich (2014)

2.1.2. Mapeamento Dinâmico

O mapeamento dinâmico consiste em múltiplos escaneamentos do ambiente combinado a informações de posição do sensor, à medida que este se movimenta pelo espaço. Torna-se evidente e direta a aplicação deste método para grandes ambientes nos quais um dispositivo em uma única posição não é capaz de mapear todo o espaço à sua volta. A robótica é certamente um dos ramos que mais se utiliza de mapeamento dinâmico, pois o sensor faz medidas em diferentes posições, movimentado por um robô que por sua vez, se desloca pelo espaço utilizando informações do mapa gerado. O tópico seguinte trata especificamente de mapeamento dinâmico em robótica, descrevendo um aspecto inerente a esta aplicação, conhecido na literatura como SLAM.

2.1.2.1. Problema SLAM

O termo SLAM é um acrônimo para *Simultaneous Localization and Mapping* (Localização e Mapeamento Simultâneos). O SLAM consiste no problema em que um robô móvel realiza o mapeamento de um ambiente desconhecido, ao mesmo tempo em que utiliza este mapa para localizar-se e navegar pelo ambiente (RIISGAARD; BLAS, 2003). Em face ao evidente conflito de precedência das atividades que o robô deve exercer, Thrun, Burgard e Fox (2000) se referem ao SLAM como um problema de natureza “ovo-galinha”.

O SLAM não se trata de algoritmo e sim de um conceito. Existem várias diferentes abordagens para o problema SLAM. Por exemplo, Yamauchi, Langley e Schultz (1998) utilizam o método de mapeamento incremental, ou seja, a cada novo movimento do robô o sensor incorpora novas informações do ambiente. Este tipo de método é rápido e permite aplicações em tempo real, no entanto, a aplicação em ambientes cíclicos pode gerar erros na construção do mapa e a consequente perda de localização do robô. Outra abordagem muita utilizada é a utilização de métodos probabilísticos. Thrun, Burgard e Fox (1998) descrevem um método de SLAM baseado no algoritmo EM *(Expectation-Maximization)*, cujo princípio sugere a construção do mapa mais próximo do real considerando a localização de todas as varreduras anteriores. A precisão do mapa gerado neste caso é bem maior, contudo, a alta demanda de processamento de dados impossibilita a aplicação em tempo real. Por fim, outro método recorrente é o que utiliza o conceito de Filtro de Kalman Estendido (EKF). Essa técnica se baseia na detecção de pontos de referência no ambiente para estimar a posição do robô.

2.1.3. Mapeamento Estático

Ao contrário do método anterior, o mapeamento estático não envolve movimentação do sensor. A princípio, o mapeamento de ambientes por este método tem como restrição o alcance máximo do sensor utilizado. Dependendo da complexidade do ambiente, o mapa gerado pode apresentar áreas sombreadas, o que é uma desvantagem. Uma forma de contornar este problema é realizar *scans* em mais de uma posição e combinar os dados obtidos em um só mapa. Esta é estratégia utilizada por Laing e Scott (2011) para gerar modelos virtuais de edifícios históricos. No exemplo descrito no trabalho, para gerar o mapa 3D de um cinema antigo, um sensor LIDAR foi posicionado em quatro diferentes posições.

Posteriormente, os dados coletados foram combinados para gerar o modelo do edifício, minimizando assim áreas não cobertas pelo sensor.

Uma vantagem deste método é que os mapas gerados estaticamente possuem alta definição (TURNER et al., 2014). Isto possibilita sua utilização em aplicações como reconhecimento de objetos. Mattausch et al. (2014) descrevem algoritmos de detecção e reconstrução de objetos de salas de escritório a partir de um sensor LIDAR. Reconhecendo similaridades entre os objetos, foi possível dividi-los em grupos e otimizar o espaço necessário para o armazenamento do modelo salvando apenas uma instância de determinado objeto.

3. Metodologia e Desenvolvimento

Esta seção trata do projeto, construção e implementação do sistema de mapeamento de ambientes. O funcionamento do sistema consiste na geração de mapas de ambientes a partir de dados obtidos por um dispositivo LIDAR 2D acoplado a um mecanismo de rotação. Conforme descrito no capítulo 2, este tipo de configuração permite a aquisição de dados tridimensionais sobre o ambiente analisado. Dentre as possíveis configurações de acoplamento descritas na seção 2.1.1.3, optou-se pela implementação da configuração de *rolling*. De acordo com o escopo definido para o projeto, o sistema de mapeamento desenvolvido deve ser estático e sua aplicação limita-se a ambientes interiores.

É pressuposto do trabalho a utilização de um sensor LIDAR da fabricante Hokuyo, modelo URG-04LX-UG01 (Figura 3.1), que é utilizado em aplicações de robótica no LASI (Laboratório de Sistemas Inteligentes), situado no Departamento de Engenharia Elétrica e Computação da Universidade de São Paulo, campus de São Carlos. O projeto do sistema foi, portanto, desenvolvido em torno da aplicação deste sensor em específico.

Figura 3.1 - Sensor Hukuyo URG-04LX-UG01
Fonte: https://www.hokuyo-aut.jp/02sensor/07scanner/urg_04lx_ug01.html

O mecanismo de rotação é composto por um motor elétrico e uma peça metálica que acopla o sensor ao eixo do motor. Além disso, um suporte fixo foi desenvolvido para manter o corpo do motor estático, de modo a minimizar efeitos de vibração durante as medições do sensor laser.

Os dados obtidos pelo sensor são transmitidos, via porta USB, a um computador que faz o processamento e armazenamento dos dados, além do controle de posição do mecanismo

de rotação. A partir da combinação dos pontos obtidos nas diversas posições de rotação do sensor, uma visualização do mapa tridimensional do ambiente é fornecida ao usuário.

Podemos dividir o projeto em duas grandes áreas: a primeira referente ao hardware, que abrange o mecanismo de rotação e o suporte de fixação do dispositivo; e a segunda relativa ao software, que abrange os processos de controle de posição e sincronização do mecanismo de rotação, aquisição dos dados do sensor, processamento dos dados, e geração da visualização do mapa tridimensional.

O diagrama apresentado na Figura 3.2 exemplifica o funcionamento geral do sistema de mapeamento tridimensional desenvolvido neste trabalho. Os detalhes de cada uma das etapas de projeto são descritos nos tópicos a seguir. Ao longo do texto, aspectos de fundamentação teórica são apresentados quando apropriado.

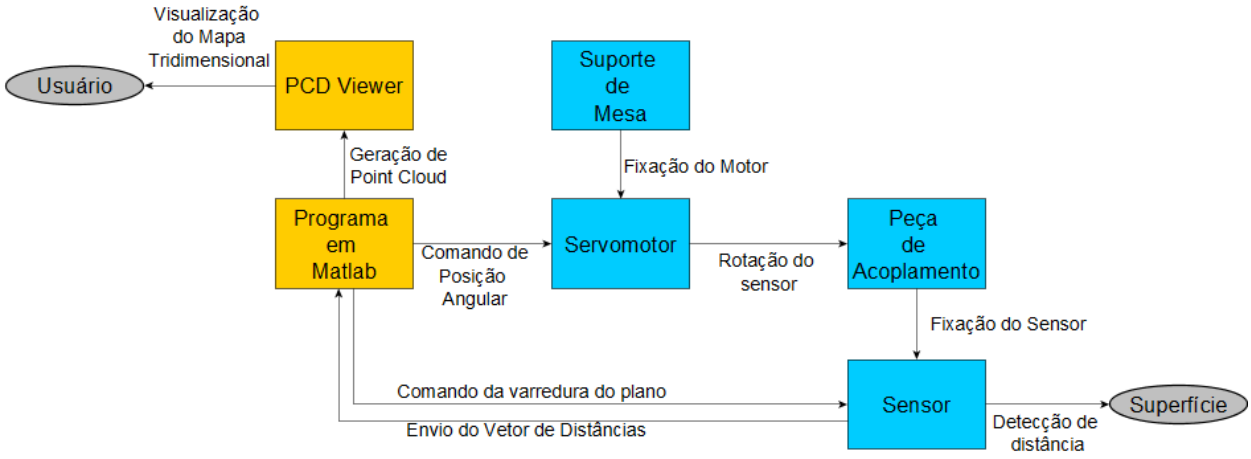


Figura 3.2 - Resumo do funcionamento geral do sistema de mapeamento tridimensional
Fonte: Autoria própria

3.1. Projeto de Hardware

3.1.1. Mecanismo de Rotação

A característica tridimensional do mapeamento é obtida através da adição de um eixo de rotação adicional ao sistema, o que torna necessário um elemento de atuação que garanta uma rotação precisa e sincronizada com a varredura do sensor. Portanto, definir um tipo e modelo de motor apropriado e que atenda aos requisitos é uma etapa fundamental do projeto. Foram considerados modelos dos tipos motor de passo, motor CC e servomotor analisando suas principais características de funcionamento, vantagens e desvantagens na aplicação neste sistema.

3.1.1.1. Análise dos tipos de motores

A primeira opção considerada é o motor de passo. Esse tipo de motor funciona a partir da conversão de pulsos elétricos em movimentos mecânicos com variações angulares discretas. O princípio de funcionamento é baseado na utilização de pares de solenoides diametralmente opostos no estator que, quando energizados, geram um campo magnético que atrai o rotor para o eixo determinado pelo par de solenoides. O rotor consiste em uma estrutura dentada. Como as bobinas são energizadas uma de cada vez, são efetuados pequenos deslocamentos denominados passos que correspondem aos alinhamentos possíveis entre os campos gerados pelas bobinas e os dentes do rotor. O motor pode ser controlado utilizando um microcontrolador e em alguns casos, modulação por largura de pulso (PWM). Um esquema de funcionamento do motor de passo é apresentado na Figura 3.3.

Uma grande vantagem dos motores de passo é que estes possuem um baixo custo comparado aos outros tipos. Além disso, devido aos seus aspectos construtivos, pouca ou nenhuma manutenção é necessária. Dependendo da resolução, definida pelo número de passos disponíveis, o motor de passo apresenta precisão considerável. Por outro lado, o motor apresenta baixa eficiência, consumindo valores altos de corrente para gerar torques relativamente baixos.

Considerando a implementação do motor de passo neste sistema específico, uma aparente vantagem seria o controle de posição em malha aberta, ou seja, sem a necessidade de um sinal de realimentação para controlar a posição do rotor. No entanto, deve ser

considerada a possibilidade de ocorrência de perda de passo do motor. A perda de passo pode ser causada em situações em que o motor atinge sua frequência de ressonância, em casos de perturbação ou devido a uma sobrecarga. Como a aplicação envolve a geração de imagens a partir de rotações do sensor que faz a varredura do ambiente, erros na posição do motor podem influenciar diretamente no resultado observado, causando distorções no mapa gerado.

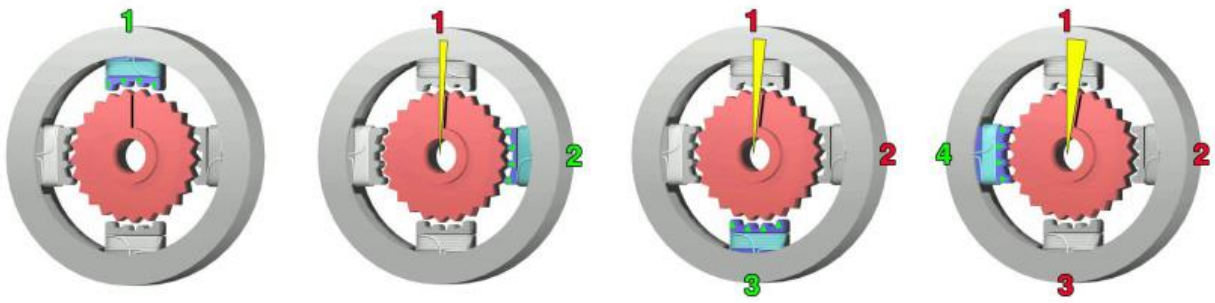


Figura 3.3 - Esquema de funcionamento do motor de passo
Fonte: Adaptado de http://pt.wikipedia.org/wiki/Motor_de_passo

O segundo tipo analisado é o motor CC. Este tipo de motor baseia-se no principio de que uma força é criada quando um condutor é percorrido por uma corrente ao estar submetido à ação de um campo magnético. Existem duas configurações de construção de motores CC, a configuração “com escovas” e a “sem escovas”, ou *brushless*. Na versão com escovas, o rotor é composto de um cilindro ao qual são enroladas espiras, em um arranjo chamado de armadura. O estator gera um campo magnético através de magnetos, que produz um torque na armadura até que os campos gerados pela armadura e estator estejam alinhados. Para produzir um torque constante no motor, é necessário manter a orientação relativa entre o campo do estator e da armadura. Para tanto, é utilizado um comutador que inverte o sentido das correntes dos enrolamentos na armadura. A aplicação de tensão elétrica no comutador é feita com a utilização de escovas; elementos de carbono que permitem a condução de corrente para partes em movimento.

Na configuração *brushless* a construção é oposta, ou seja, os enrolamentos estão situados no estator, e os magnetos no rotor. Como o próprio nome indica, o motor CC *brushless* não possui escovas e também não utiliza comutador. O sentido de rotação do rotor é mantido devido à aplicação de corrente alternada nas espiras dos estator. A forma de onda da corrente pode ser trapezoidal ou senoidal e é gerada por um driver eletrônico que faz a inversão da corrente a partir de um sensor de efeito Hall.

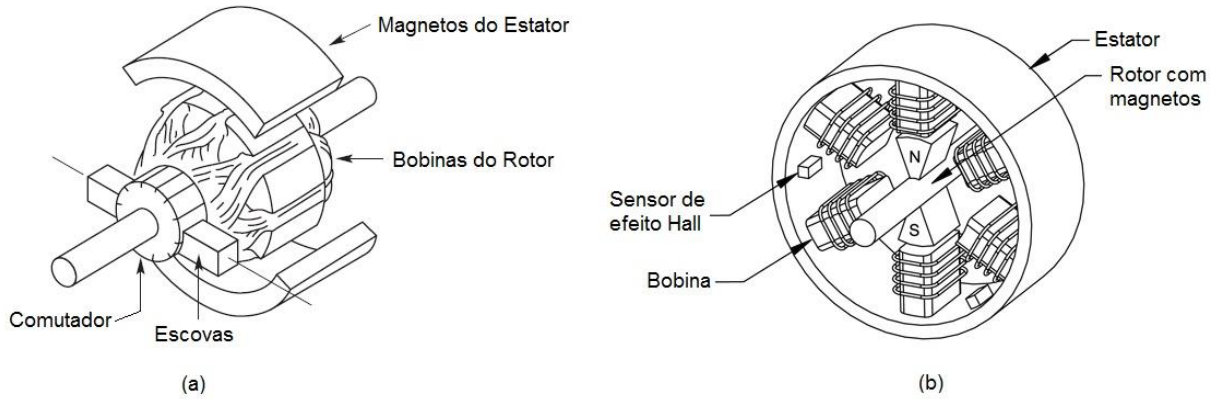


Figura 3.4 - Esquema do motor CC (a) configuração "com escovas"; e (b) configuração brushless
Fonte: Adaptado de http://www.cordlessimpactdriverhq.com/impact-drivers/impact-drivers-brushless-motors

Os motores CC são ótimas opções em aplicações que demandam controle preciso de velocidade. Como a velocidade de rotação é diretamente proporcional à tensão aplicada na armadura, o controle de velocidade e torque é relativamente simples. No caso do motor CC com escovas, um ponto negativo é que, após muitos ciclos, o comutador e as escovas se desgastam e precisam de manutenção. A configuração *brushless* não apresenta este problema.

Um fator limitante dos motores CC em geral para aplicação no sistema de mapeamento é a baixa precisão de posição angular. Torna-se inviável obter um controle preciso de posição em malha aberta e, portanto, se faz necessário o uso de um sensor de posição, tal qual um encoder, para realimentar o sistema de modo que o rotor convirja para a posição desejada.

Por fim, foram analisadas as características de servomotores. Este tipo de motor é ideal para aplicações que necessitem de controle exato da posição angular. A alta precisão é obtida graças à implementação de um sistema de malha fechada incorporado à carcaça do motor. A estrutura de um servomotor é composta basicamente por: um motor (na maioria dos casos do tipo CC, no entanto também são usados modelos do tipo CA), um sensor de posição (potenciômetro ou encoder), um conjunto de engrenagens e um circuito de controle.

Neste arranjo, o sensor de posição gera um sinal de realimentação referente à posição atual do eixo do motor, que é comparado no circuito de controle com um sinal de referência contendo a posição angular desejada. A diferença entre estes sinais configura o erro que tem de ser zerado para que o motor alcance a posição requerida. Na maioria dos casos o circuito de controle trata-se de um controlador PID. Em servomotores mais simples, potenciômetros

são utilizados com sensor de posição, fornecendo o sinal de controle através da resistência elétrica medida entre seus terminais. Em modelos de altíssima precisão, são utilizados encoders óticos para o sensoriamento da posição. O conjunto de engrenagens compõe uma caixa de redução e é utilizado para amplificar o torque do motor. O esquema de funcionamento de um servomotor é exemplificado no diagrama da Figura 3.5.

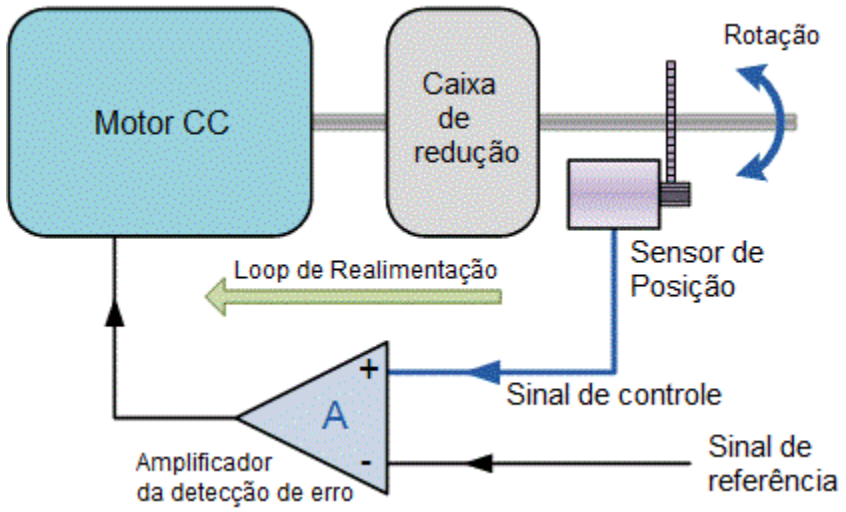


Figura 3.5 - Diagrama de funcionamento do servomotor
Fonte: Adaptado de http://www.electronics-tutorials.ws/io/io_7.html

Como já mencionado, uma das grandes vantagens do servomotor é o controle preciso de posição e velocidade graças ao sensor de posição que fornece o sinal de realimentação ao sistema, fazendo que este opere em malha fechada. Esse atributo é altamente desejado na aplicação para o mecanismo de rotação do sistema de mapeamento, pois a medida da posição angular do motor deve ser usada para sincronização com as leituras feitas pelo sensor LIDAR e o cálculo das coordenadas (x, y, z) da superfície varrida.

Evidentemente, por conter elementos de controle que possibilitam uma acurácia maior, o servomotor possui um custo mais elevado do que os outros tipos de motor. Além disso, devido à complexidade de sua construção e componentes, servomotores demandam um ajuste fino e manutenção com mais frequência.

3.1.1.2. Requisitos do Motor

Além de analisar o tipo de motor mais adequado para aplicação no sistema, devem ser considerados os requisitos que o modelo escolhido deve atender, como o torque necessário e a interface de comunicação utilizada para integração com o sensor.

O torque necessário tem relação direta com as características de geometria e massa da carga a ser acoplada ao eixo do motor. Um conceito importante para determinar o torque necessário para movimentar o sensor trata-se do momento de inércia. Esta é a medida física que determina a quantidade de torque que deve ser aplicado para alterar o momento angular de um corpo rotacionando em torno de um eixo específico. O momento de inércia de um corpo rígido C é calculado através da seguinte expressão:

$$I = \int_C R^2 dm \tag{3.1}$$

onde dm refere-se ao elemento de massa dado pelo fator densidade multiplicado por uma diferencial apropriada (volume, área ou comprimento). O termo R é relativo à distância perpendicular entre o elemento de massa e o eixo de rotação.

As especificações completas do sensor Hokuyo são apresentadas no Apêndice E. A massa do sensor é de aproximadamente 160 g e deve ser adicionada à massa da peça de acoplamento entre sensor e motor, que é estimada em 40 g. De modo a simplificar o cálculo de momento de inércia do sensor, a sua geometria foi aproximada à forma do paralelepípedo ao qual o mesmo está inscrito, como indica a Figura 3.6a.

Um elemento de massa do paralelepípedo representado na Figura 3.6b pode ser descrito por:

$$dm = \delta dxdydz \tag{3.2}$$

onde δ é a densidade do elemento incremental. A distância R entre o elemento de massa e o eixo x, em torno do qual o corpo gira, é dada por:

$$R = \sqrt{y^2 + z^2} \tag{3.3}$$

Aplicando em (3.1), obtemos a expressão que define o momento de inércia do paralelepípedo em relação ao eixo x, com a, b e c como na Figura 3.6a.

$$I_x = \int_{-\frac{c}{2}}^{\frac{c}{2}} \int_{-\frac{b}{2}}^{\frac{b}{2}} \int_{0}^{a} \delta(y^2 + z^2) dx dy dz \tag{3.4}$$

Resolvendo (3.4), chegamos à seguinte equação

$$I_x = \frac{\delta\, abc}{12}(b^2 + c^2) \tag{3.5}$$

Considerando distribuição uniforme de massa, temos que

$$\delta = \frac{M}{abc} \tag{3.6}$$

onde M é massa total do corpo. Substituindo em (3.5) chegamos ao seguinte resultado:

$$I_x = \frac{M}{12}(b^2 + c^2) \tag{3.7}$$

Portanto, sabendo que $M = 0{,}2\ Kg$, $b = 0.05\ m$ e $c = 0.07\ m$, substituindo estes valores em (3.7) obtemos o valor numérico do momento de inércia do corpo.

$$I_x = 1{,}233 \cdot 10^{-4}\ Kg.m^2 \tag{3.8}$$

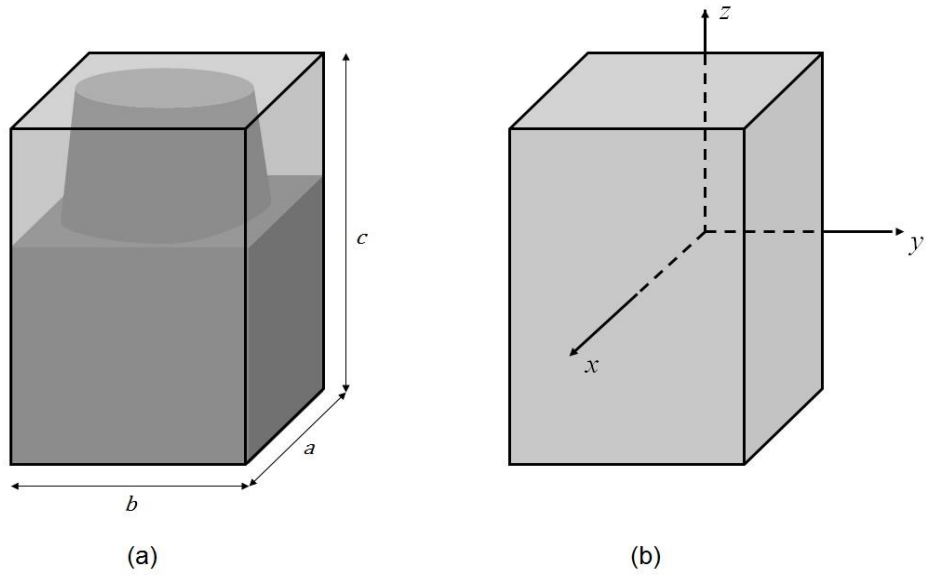


Figura 3.6(a) Dimensões do paralelepípedo utilizado para o cálculo de momento de inércia; (b) orientação do paralelepípedo

Fonte: Autoria própria

O torque necessário do motor é determinado a partir do cálculo dos torques de carga e aceleração. (ORIENTAL MOTOR, 2012) O torque de carga T_L é dependente das forças externas que atuam sobre a carga e o raio definido pela distância entre o eixo de rotação e a extremidade da carga. Como neste caso a única força externa atuante sobre o sensor é o peso devido à gravidade, a expressão que calcula o torque de carga é:

$$T_L = M.g.r \tag{3.9}$$

Com $g = 9{,}8067\ m/s^2$ e $r = 35.10^{-3}\ m$, obtém-se:

$$T_L = 7{,}0.10^{-3}\ Nm \tag{3.10}$$

Por sua vez, o torque de aceleração T_A depende do momento de inércia da carga J_L e o momento de inércia do rotor J_0, além da frequência de operação f estimada e a resolução angular do motor θ_s. A expressão que calcula o torque de aceleração é, portanto:

$$T_A = (J_0 + J_L) \cdot \frac{\pi \cdot \theta_S}{180} \cdot f^2 \tag{3.11}$$

O valor do momento de inércia da carga foi obtido em (3.8). A frequência de operação f e a resolução θ_s são estimados em 10 Hz e 0,36°, respectivamente. Portanto, o valor de T_Aé dado por:

$$T_A = 0{,}628J_0 + 7{,}75 \cdot 10^{-5}\ Nm \tag{3.12}$$

Tendo definido os valores dos torques de carga e aceleração, pode-se estimar o torque necessário do motor a partir da seguinte expressão:

$$T_M = (T_L + T_A)K_s \tag{3.13}$$

Onde K_s é o fator de segurança e, tipicamente, assume valores entre 1,5 e 2. Com $K_s = 2$, temos, portanto o valor estimado do torque necessário do motor:

$$T_M = 0{,}013 + 1{,}25J_0\ Nm \tag{3.14}$$

Portanto, sendo J_0 o momento de inércia do rotor de um motor considerado para essa aplicação, o torque máximo fornecido por este motor deve ser superior ao valor calculado em (3.14).

3.1.1.3. Escolha do Motor

Baseado na análise sobre os tipos de motor descrita no tópico 3.1.1.1, decidiu-se que o modelo mais apropriado para a utilização no sistema de mapeamento é o servomotor. Chegou-se a esta conclusão analisando parâmetros, principalmente, de desempenho, custo e simplicidade de implementação. Os modelos de motor de passo e motor CC apresentam custo mais baixo, porém, para atingir um desempenho semelhante ao de um servomotor, seria necessário adicionar um sensor de posição e um controlador; o que, além de encarecer essas opções, tornaria mais complexa a utilização das mesmas. Apesar de o servomotor ter um custo mais elevado, este já possui os elementos necessários para atuação em malha fechada, o que é uma característica desejada neste sistema.

Dos modelos de servomotor disponíveis no mercado, optou-se pelo Dynamixel AX-12A, da fabricante Robotis (Figura 3.7). Este modelo possui todas as funcionalidades de um servomotor típico, como sensor de realimentação (é utilizado um potenciômetro), caixa de redução e componente eletrônicos de controle. Estes elementos são encapsulados em um invólucro compacto, feito em plástico de engenharia, que pesa em torno de 55 g. Condições indesejadas como temperatura, tensão e torque fora da região normal de operação podem ser detectadas, sinalizadas (através de um LED) e contornadas automaticamente. O torque máximo fornecido por este modelo é de 1,5 Nm, valor bem superior ao torque necessário estimado em 3.1.1.2; o que garante a aplicabilidade do motor para este sistema. O Dynamixel é destinado a aplicações em robótica e foi utilizado em projetos semelhantes como em Sheh e Jamali (2006).

Figura 3.7 - Servomotor Dynamixel AX-12A
Fonte: Manual do produto AX-12A - http://support.robotis.com/en/techsupport_eng.htm

Uma grande vantagem do Dynamixel é que este possui uma interface de comunicação USB que permite o controle do motor por computador em tempo real. A interface é feita pelo componente USB2Dynamixel (Figura 3.8), que pode ser conectado tanto a uma porta serial, como a uma porta USB do computador. A conexão entre o USB2Dynamixel e o motor é feita através de um cabo de três pinos utilizando comunicação TTL. A interface USB, no entanto, não é suficiente para alimentação elétrica do motor. Portanto deve ser utilizada uma fonte CC ou bateria externa de 12V, que possa suprir correntes de até 1A. Neste trabalho a alimentação do motor foi fornecida por uma fonte chaveada de 12V, com corrente máxima de 2A.

Figura 3.8 - Interface USB entre computador e motor USB2Dynamixel
Fonte: Manual do produto USB2Dynamixel - http://support.robotis.com/en/techsupport_eng.htm

3.1.1.4. Peça de Acoplamento

Esta peça tem por objetivo acoplar o sensor ao eixo do motor. A escolha do tipo de rotação que o motor aplica no sensor – *pitching, yawing* ou *rolling* – é essencial para o projeto desta peça. Neste trabalho o método aplicado é o de *rolling*. Este método proporciona vantagens em relação aos demais, pois o motor pode ser posicionado na área que não é varrida pelo sensor, o que otimiza o espaço ocupado pelo conjunto motor-sensor. Além disso, nesta configuração o sensor fica mais próximo ao eixo do motor, o que minimiza o torque e, portanto torna o sistema mais eficiente.

Um ponto importante é definir em torno de que eixo o sensor será rotacionado; a princípio duas possibilidades são consideradas. Na primeira, o eixo de rotação está contido no plano de varredura do sensor, como indica a Figura 3.9a. Esta configuração apresenta muitas vantagens em relação ao cálculo das coordenadas obtidas pelo sensor, uma vez que não são

necessárias correções e as medições em cada plano possuem sempre um ponto em comum; aquele correspondente ao ponto central da varredura em qualquer das posições que o sensor esteja. Por outro lado, nesta configuração o centro geométrico do sensor está mais distante do eixo de rotação do motor. Logo, o torque necessário será maior e o sensor ocupará um raio maior durante sua operação.

Na segunda configuração, o sensor é rotacionado em torno de seu centro geométrico, como pode ser visto na Figura 3.9b. Neste caso, as vantagens são exatamente opostas às da configuração anterior. Com o acoplamento do eixo do motor mais próximo ao centro geométrico do sensor, o torque necessário é menor e o conjunto motor-sensor ocupa menos espaço. No entanto, são necessários cálculos complementares para compensar a distância entre o centro geométrico do sensor e plano de varredura. Se esta correção não é feita, cria-se uma área circular à frente do sensor não pode ser mapeada; o que é indesejável.

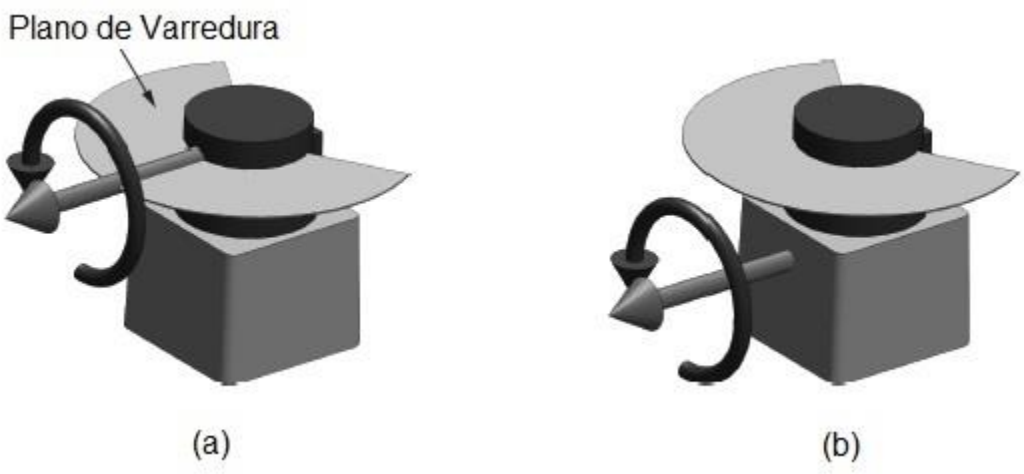


Figura 3.9 - Eixo de rotação do sensor localizado (a) no plano de varredura; e (b) no centro geométrico
Fonte: Adaptado de Tâche et al. (2011)

Neste trabalho, optou-se por projetar uma peça de acoplamento que implementasse a segunda configuração descrita, ou seja, com o sensor girando em torno de seu centro geométrico. Os cálculos necessários para a compensação da distância entre o centro geométrico e o plano de varredura são descritos na seção 3.2.4. A peça projetada foi feita em alumínio, de modo a conferir baixo peso e resistência mecânica suficiente ao sistema. Uma das faces da peça é fixada ao eixo do motor e a outra sustenta o sensor. Uma foto da peça já usinada é mostrada na Figura 3.10. O diagrama utilizado para a fabricação da peça é apresentado no Apêndice A.

Figura 3.10 - Peça de acoplamento entre o motor e o sensor
Fonte: Autoria própria

3.1.2. Suporte de Fixação

Este suporte foi projetado para garantir que o mecanismo de rotação esteja alheio a perturbações externas e para que a reação ao torque aplicado pelo motor não cause movimentação indesejadas durante as medições. Para tanto, foi adaptado um suporte tipo grampo que pode ser fixado em mesas ou outras superfícies. Uma peça de alumínio que conecta este suporte de mesa ao corpo do motor foi projetada. O diagrama desta peça é apresentado no Apêndice B. O sistema completo, com suporte de fixação, mecanismo de rotação e sensor, é apresentado na Figura 3.11.

Figura 3.11 - Montagem do sistema completo
Fonte: Autoria própria

3.2. Projeto de Software

Esta seção descreve os procedimentos e fundamentos teóricos envolvidos na elaboração dos métodos computacionais utilizados no sistema de mapeamento. A plataforma de desenvolvimento escolhida foi o MATLAB (Matrix Laboratory). O MATLAB oferece uma linguagem de alto nível e muitas funções pré-definidas que facilitam a escrita do programa. Além disso, oferece um ambiente intuitivo de programação onde é possível testar e validar rapidamente o código escrito. No caso deste sistema em especifico, a interface serial com dispositivos externos é um requisito importante. O MATLAB possui uma toolbox dedicada a este tipo de aplicação. Através da função *serial*, é possível configurar rapidamente a comunicação serial entre o computador e dispositivo. Entre outras vantagens, esta facilidade reforça a escolha desta plataforma.

A seguir são detalhados os aspectos necessários para o desenvolvimento do software do sistema. O código que implementa os conceitos aqui descritos é apresentado na íntegra no Apêndice F.

3.2.1. Características do sensor Hokuyo

Para uma correta utilização e obtenção dos resultados desejados, é essencial o entendimento do modo de funcionamento do sensor. Um estudo detalhado sobre o modelo de sensor utilizado neste trabalho pode ser encontrado na caracterização feita por Okubo, Ye & Borenstein (2009). Neste tópico são descritos os princípios fundamentais de funcionamento do sensor.

O Hokuyo URG-04LX-UG01 é um sensor do tipo LIDAR, da categoria AMCW[2]. Um laser emite um feixe de luz infravermelho de comprimento de onda 785 nm e classe de segurança 1[3]. A direção do feixe é alterada por um espelho rotativo, atinge a superfície do objeto e é refletida de volta ao sensor. O espelho rotativo mais uma vez altera a direção do feixe refletido que é então detectado por um fotodiodo. Através da comparação entre as fases das ondas recebida e

[2] *Amplitude Modulated Continuous Wave*, em português: Onda Contínua Modulada em Amplitude

[3] A radiação laser pode ser prejudicial aos olhos. Os dispositivos laser possuem classificação definidas pela IEC (*International Electrotechnical Comission)* referente às condições de utilização segura do equipamento. As classes são, em ordem crescente de criticalidade, 1, 1M, 2, 2M, 3R, 3B e 4. A classe 1, portanto, é a menos crítica, não gerando riscos sob condições normais de utilização (http://www.rp-photonics.com/laser_safety.html)

refletida, calcula-se a distância entre o sensor e o objeto (como descrito na seção 2.1.1.3). O princípio de operação do sensor Hokuyo é representado na Figura 3.12.

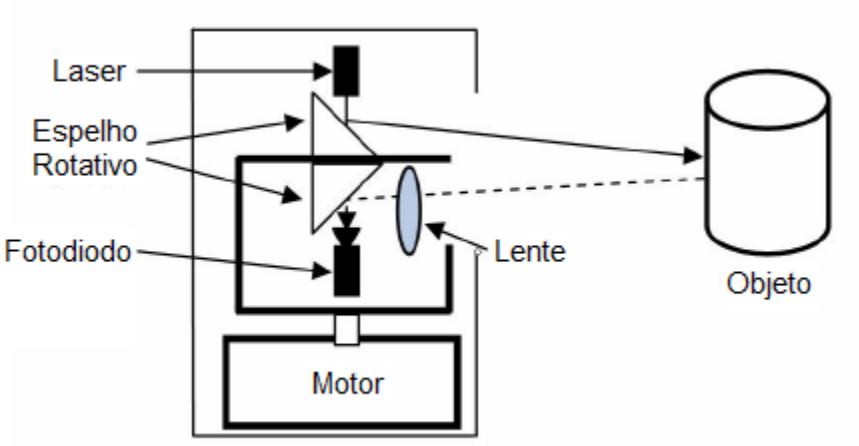


Figura 3.12 - Esquema de funcionamento do sensor Hokuyo
Fonte: Adaptado de Okubo, Ye e Borenstein (2009)

O espelho é rotacionado por um motor, que possibilita uma varredura no plano de um semicírculo de 240°, com resolução de 0,36° (360°/1024) e raio máximo de 4000 mm (vide Figura 3.13). Para medições entre 20 e 1000 mm, a precisão é de ±30 mm. Já distâncias maiores que 1000 mm apresentam erro máximo de ±3% da medida. A frequência de rotação do motor é de aproximadamente 600 rpm, o que garante uma taxa de varredura de 100 ms. A varredura é feita no sentido anti-horário.

As medições são transmitidas a um *host* através de uma interface USB com taxa de transmissão de 9 Mbps. A tensão de 5 V necessária para alimentar o sensor também é fornecida via USB, porém utilizando um cabo distinto ao de comunicação de dados. (Hokuyo Specifications, 2009)

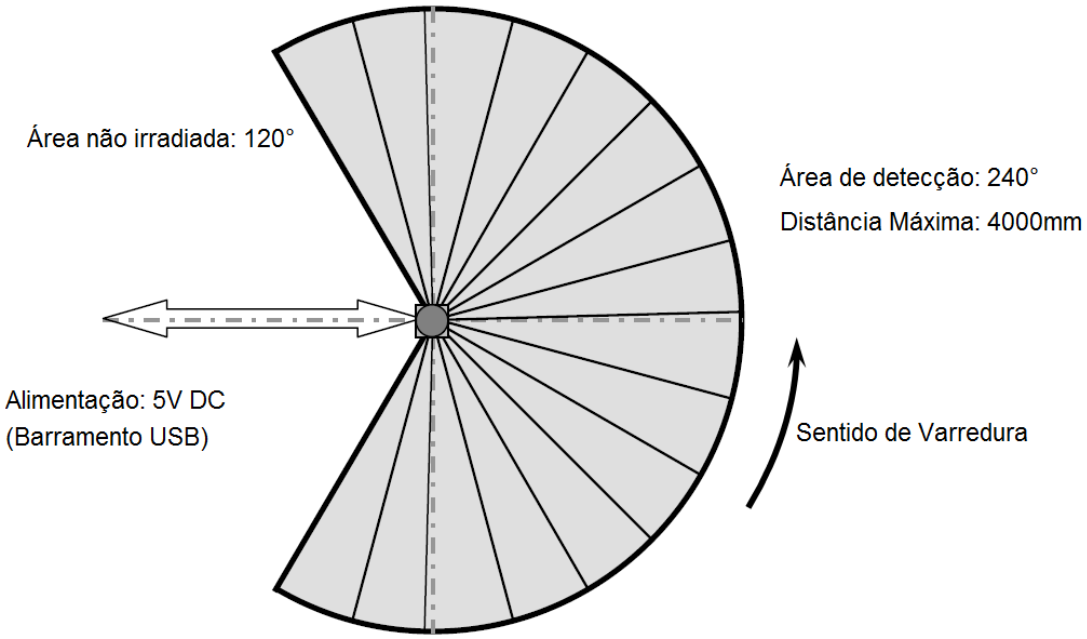


Figura 3.13 - Área de cobertura do sensor Hokuyo
Fonte: Adaptado de URG-04LX-UG01 Specifications (2009)

3.2.2. Aquisição de dados

A comunicação entre o sensor e o *host* (no caso deste trabalho um notebook) é feita utilizando o protocolo de comunicação SCIP 2.0. Este protocolo foi desenvolvido pelo grupo de pesquisa do Intelligent Robot Laboratory da Universidade de Tsukuba com o intuito de prover uma interface eficiente e flexível para interface de sensores em aplicação robóticas.

O protocolo SCIP (*Scanning sensor Command Interface Protocol*) utiliza codificação para reduzir o tempo de transmissão entre sensor e *host*. A codificação baseia-se na conversão de valores binários em caracteres ASCII. Dois modos de codificação são utilizados: dados com tamanho máximo de 12 bits são descritos por dois caracteres ASCII, enquanto que dados até 18 bits são convertidos em três caracteres. Exemplos de aplicação dos algoritmos de codificação para dois e três caracteres são apresentados nas Figura 3.14 e Figura 3.15, respectivamente.

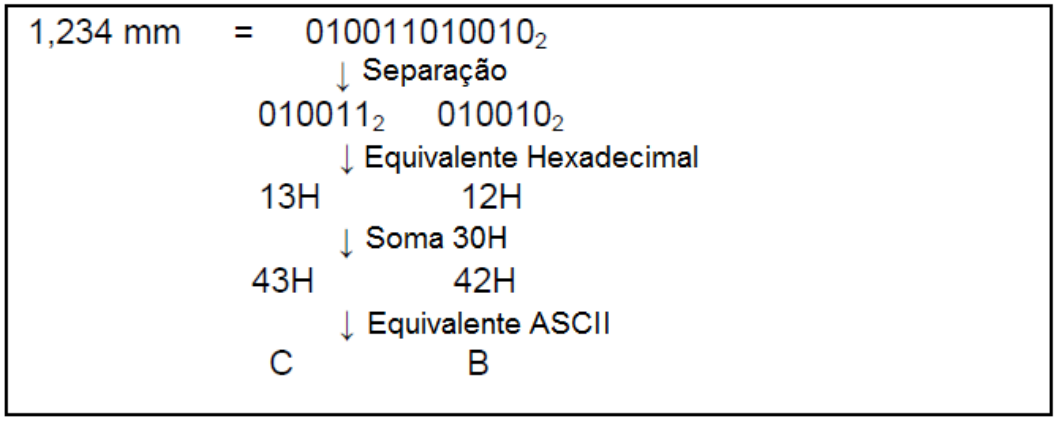


Figura 3.14 - Algoritmo de codificação de 2 caracteres (12 bits)
Fonte: Adaptado de Hokuyo - Specification for SCIP2.0 Standard (2006)

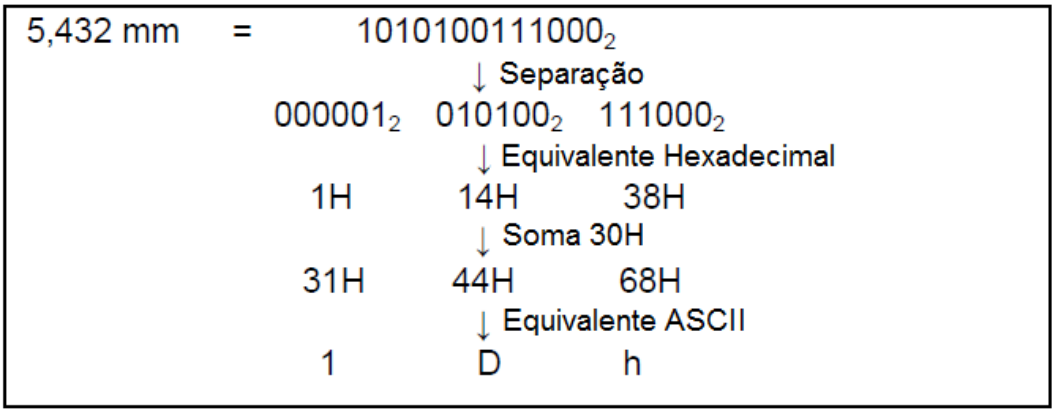


Figura 3.15 - Algoritmo de codificação de 3 caracteres (18 bits)
Fonte: Adaptado de Hokuyo - Specification for SCIP2.0 Standard (2006)

Os valores de medidas e outros parâmetros, como mensagens de erro, produzidos pelo sensor são transmitidos codificados. Portanto antes de serem processados, os dados devem ser decodificados no *host*. O processo de decodificação nada mais é do que operação inversa do algoritmo de codificação. As Figura 3.16 e Figura 3.17 apresentam os algoritmos de decodificação para dois e três caracteres, respectivamente.

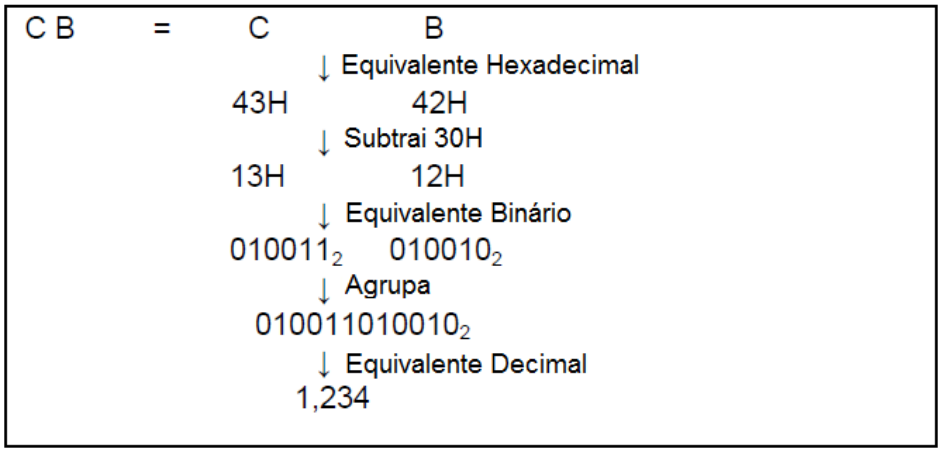


Figura 3.16 - Algoritmo de decodificação de 2 caracteres
Fonte: Adaptado de Hokuyo - Specification for SCIP2.0 Standard (2006)

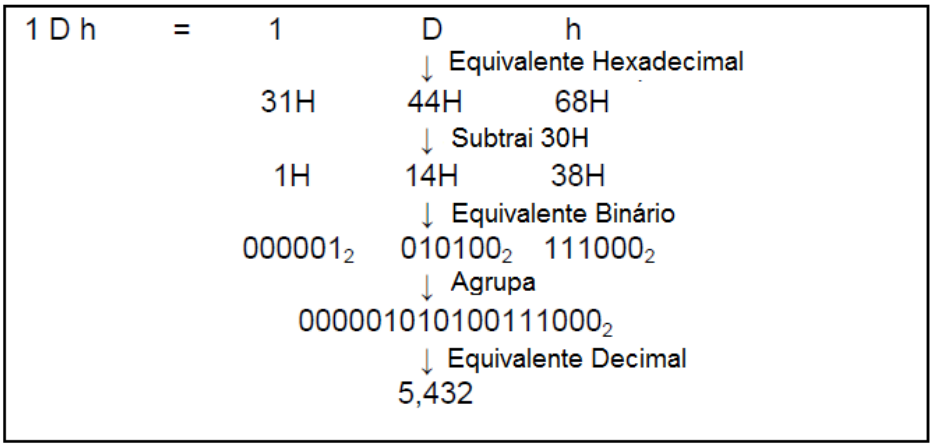


Figura 3.17 - Algoritmo de decodificação de 3 caracteres
Fonte: Adaptado de Hokuyo - Specification for SCIP2.0 Standard (2006)

O formato de comunicação entre sensor e *host* em SCIP consiste em um conjunto de comandos pré-definidos. Um comando é uma palavra de dados enviada pelo *host* ao sensor. O sensor, por sua vez, após receber o comando e gerar o dado requisitado, transmite uma resposta ao *host*. As palavras genéricas de comando e resposta são exemplificadas na Figura 3.18. Existem 13 tipos de comandos pré-definidos em SCIP.

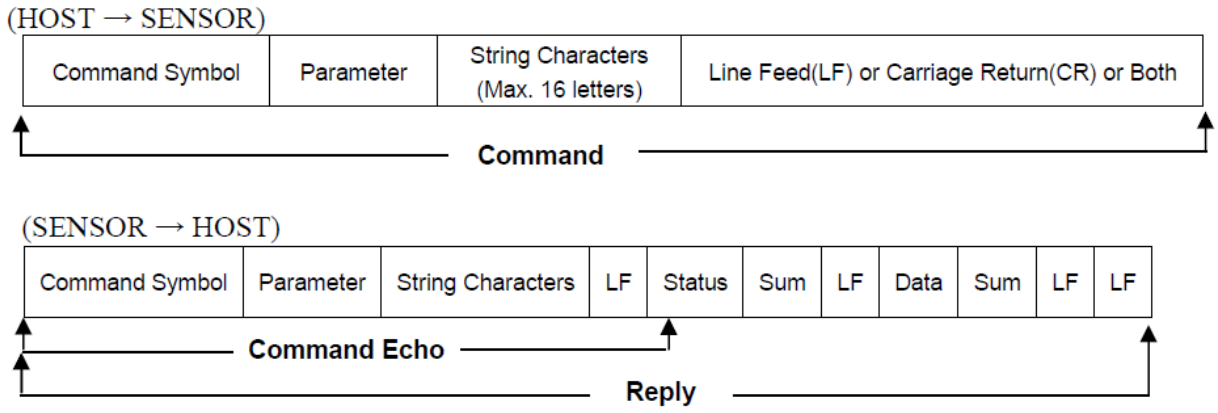


Figura 3.18 - Estrutura das palavras de comando e resposta em SCIP 2.0
Fonte: Adaptado de Hokuyo - Specification for SCIP2.0 Standard (2006)

Para aquisição do vetor de distâncias medidas em um determinado plano, foi utilizado o comando GD. Este comando retorna ao *host* os dados da última medição feita pelo sensor. Na palavra de comando, a seção de parâmetro é preenchida com os valores de passo inicial e final, referentes à região que é coberta pelo feixe do sensor. Um exemplo do posicionamento de passos inicial e final é apresentado na Figura 3.19. Para essa aplicação foram utilizados os valores de passos que garantem a cobertura máxima do sensor (240°). Neste caso, o valor do passo inicial é 44 e o passo final corresponde a 725. Os valores de distância medidos para cada passo são alocados na seção de dados da palavra de resposta e então transmitidos ao *host*. Desta forma, em cada varredura do sensor, o *host* recebe uma resposta que, quando decodificada, produz um vetor de 682 posições.

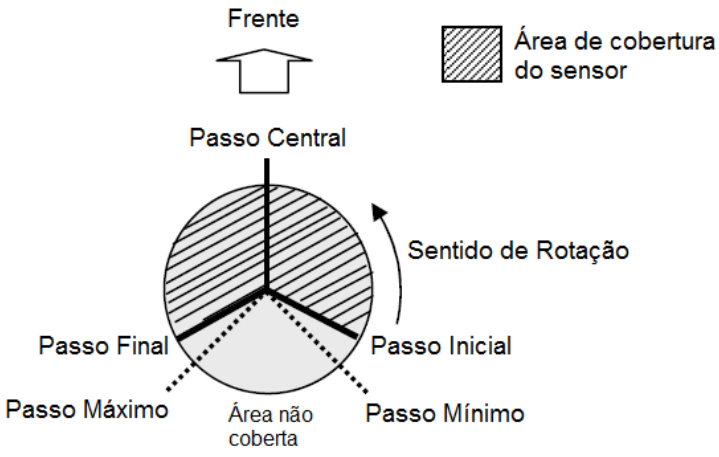


Figura 3.19 - Diagrama com as posições de passo de varredura do sensor Hokuyo
Fonte: Adaptado de Hokuyo - Specification for SCIP2.0 Standard (2006)

3.2.3. Sincronização do sistema

Uma vez obtido o vetor de distâncias para um plano, deseja-se realizar o mesmo processo para outros planos, de modo a obter a representação tridimensional do ambiente. Para tanto, o sensor é gradualmente reposicionado pelo motor cobrindo um ângulo de 180°, suficiente para mapear completamente superfícies dentro do raio de alcance do sensor. É fundamental, portanto, que o motor posicione o sensor com precisão e com uma frequência apropriada.

Assim como o sensor Hokuyo, o motor Dynamixel AX12-A se comunica com um *host* através de um padrão pré-definido. O protocolo utilizado nessa interface é o de comunicação serial assíncrona half duplex, com palavras de 8 bits de dados, 1 bit de finalização e sem paridade. As palavras enviadas pelo *host* ao motor constituem o pacote de instruções, enquanto que as palavras recebidas pelo *host* vindas do motor formam o pacote de status. Na Figura 3.20 são apresentas as estruturas genéricas de pacotes de instrução e status. As operações realizadas pelos pacotes de instrução e status se resumem a uma tabela de controle do motor. Esta tabela estabelece endereços da memória interna do motor para cada parâmetro do mesmo. O conteúdo das posições de memória pode ser lido para se obter o status de um determinado parâmetro e pode-se escrever em posições de memória referentes a parâmetros controláveis para dar algum comando ao motor. Operações de escrita são feitas utilizando uma palavra de instrução, enquanto que a leitura é feita utilizando a palavra de status. A tabela de controle do Dynamixel AX-12A é apresentada no Apêndice D.

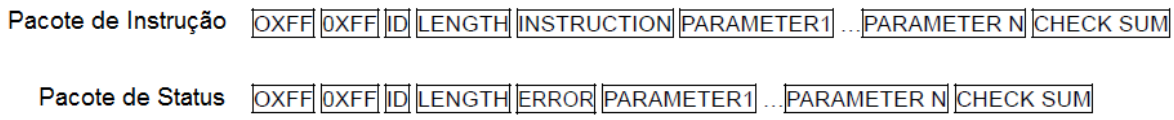


Figura 3.20 - Estruturas dos pacotes de instrução e status dos motor Dynamixel AX-12A
Fonte: Adaptado de Manual do produto AX-12A - http://support.robotis.com/en/techsupport_eng.htm

Conforme mencionado na seção 3.1.1.1, um servomotor contém elementos que o permitem operar em malha fechada; este é o caso do Dynamixel AX12-A. Um dos parâmetros cujo controle utiliza o sinal de realimentação é a posição angular desejada ou *Goal Position.* Este parâmetro permite que uma posição angular seja definida pelo *host,* escrita no endereço de memória correspondente, fazendo que o eixo do motor se desloque com erro minimizado. Isso garante que o requisito de precisão da posição angular do eixo do motor seja cumprido,

além de tornar mais simples o controle do motor. A Figura 3.21 exemplifica o endereçamento do parâmetro *Goal Position.* É importante notar que a seção entre os ângulos 300° e 360° não é endereçável. A tabela de controle define que o parâmetro *Goal Position* é alocado nos endereços 0x1E (byte menos significativo) e 0x1F (byte mais significativo). No exemplo da Figura 3.21, quando os valores 0xff e 0x03 são atribuídos aos endereços 0x1E e Ox1F respectivamente, o eixo do motor é movido para a posição angular de 300°.

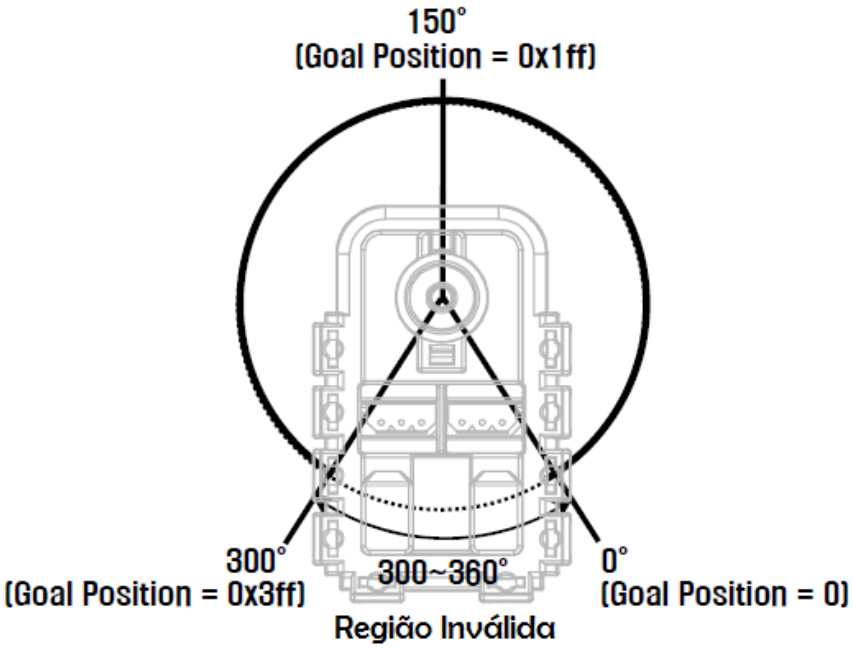


Figura 3.21 - Diagrama das posições angulares endereçáveis do motor Dynamixel AX-12ª
Fonte: Adaptado de Manual do produto AX-12A - http://support.robotis.com/en/techsupport_eng.htm

A sincronização entre a rotação do eixo do motor e a varredura do sensor pode ser implementada incrementando a posição angular do eixo do motor ao final de cada varredura. Como um pulso de sincronização não é disponível neste modelo de sensor, utiliza-se o artifício de definir um tempo limite para a comunicação serial. O parâmetro Timeout na configuração da interface serial entre sensor e *host* no Matlab exerce essa função. Ao se atingir o tempo limite definido por Timeout, a comunicação serial é interrompida e um novo comando para incrementar a posição angular do sensor é transmitido.

A posição angular do motor tem uma resolução de 0,29° (300°/1024). Para se obter uma representação completa do sistema, sem repetir desnecessariamente medidas, o eixo do motor varia entre as posições -90° e +90°, totalizando um movimento de 180°. Neste intervalo,

há 614 posições angulares (ou passos) possíveis. Portanto, em sua resolução máxima, o sistema de mapeamento produz 418748 pontos (614 posições angulares x 682 medidas por plano).

3.2.4. Processamento dos dados

Esta etapa do projeto consiste em determinar as coordenadas *(x, y, z)* de um ponto da superfície com informações atuais da posição angular do sensor, definida pelo mecanismo de rotação, e a posição angular do feixe emitido pelo sensor. Tanto o motor como o feixe de varredura do sensor tem suas posições angulares definidas em passos. Portanto, para que se possa realizar qualquer operação matemática com estes parâmetros, os mesmos devem ser primeiramente convertidos em graus. Como o deslocamento angular entre dois passos é constante, a conversão de passos para graus pode ser feita através de funções lineares. Definindo β como o ângulo descrito pelo feixe de varredura e θ o ângulo de rotação do sensor, como na Figura 3.22, temos as seguintes expressões:

$$\beta_i = \frac{240}{681}\vartheta_i - \frac{81960}{681} \tag{3.15}$$

$$\theta_i = \frac{300}{1024}k_i - 150 \tag{3.16}$$

onde ϑ_i se refere ao i-ésimo elemento do vetor de varredura e k_i é o valor do i-ésimo passo do eixo do motor.

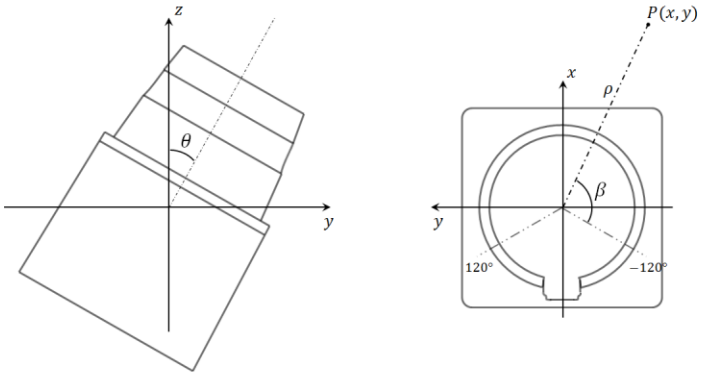


Figura 3.22 - Orientação do sensor Hokuyo e definição dos ângulos θ e β
Fonte: Autoria própria

A determinação das coordenadas tridimensionais no espaço envolve o problema de representar um ponto em um determinado sistema de coordenadas em relação a outro sistema. Essa operação é realizada determinando a matriz de transformação homogênea que relaciona as representações nos dois sistemas de coordenadas.

Uma matriz de transformação homogênea é a representação de um movimento rígido de um corpo. O movimento rígido é a combinação dos movimentos de rotação e translação puros. As matrizes de rotação pertencem a um grupo denominado $SO(n)$, enquanto que o vetor de translação pertence a $\mathbb{R}^n$. Portanto, dado um ponto p em um sistema de coordenadas {1}, as coordenadas deste ponto em um sistema de coordenadas {0} são obtidas através da expressão:

$$\begin{bmatrix} p^0 \\ 1 \end{bmatrix} = H_1^0 \begin{bmatrix} p^1 \\ 1 \end{bmatrix} \tag{3.17}$$

Na qual H_1^0 é a transformada homogênea, que é da seguinte forma:

$$H = \begin{bmatrix} R & d \\ 0 & 1 \end{bmatrix} ; R \in SO(3), d \in \mathbb{R}^3 \tag{3.18}$$

O grupo de transformadas homogêneas básicas, que fazem parte do conjunto $SE(3)$, é apresentado abaixo. As transformadas se referem a translações e rotações puras em relação aos eixos *x*, *y* e *z*.

$$Trans_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad Rot_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & sen\alpha & 0 \\ 0 & -sen\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.19}$$

$$Trans_{y,b} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad Rot_{y,\beta} = \begin{bmatrix} \cos\beta & 0 & sen\beta & 0 \\ 0 & 1 & 0 & 0 \\ -sen\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.20}$$

$$Trans_{z,c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}; \qquad Rot_{z,\gamma} = \begin{bmatrix} \cos\gamma & sen\gamma & 0 & 0 \\ -sen\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.21}$$

Matrizes de transformação homogênea podem também ser combinadas para descrever uma sequência de movimentos rígidos. No entanto, deve ser observada a regra de composição de transformadas homogêneas. Segundo a regra, dada uma matriz de transformação homogênea H_1^0 relacionando dois sistemas de coordenadas, se for aplicado um movimento rígido adicional representado por $H \in SE(3)$ em relação ao sistema de coordenadas atual, define-se a composição das transformadas homogêneas como:

$$H_2^0 = H_1^0 H \tag{3.22}$$

ou seja, realiza-se uma pós-multiplicação. Caso o movimento rígido adicional seja feito em relação ao sistema de coordenadas fixo, a operação realizada é de pré-multiplicação, como abaixo

$$H_2^0 = H H_1^0 \tag{3.23}$$

Os conceitos de transformadas homogêneas podem ser utilizados, portanto, para que as coordenadas dos pontos em cada plano varrido pelo sensor sejam referenciadas a um mesmo sistema de coordenadas, formando assim o mapa tridimensional desejado. O sistema de coordenadas {0} escolhido como referência tem sua origem no centro geométrico do sensor, com o eixo x coincidente ao eixo do motor e sentido apontando para frente do sensor. O eixo z é definido apontando para cima e o sentido do eixo y é obtido a partir da regra da mão direita (Regra de Fleming).

Um sistema de coordenadas {n} tem sua origem contida no plano de varredura do sensor, com eixo x apontando para frente do sensor, o eixo z com sentido para a parte superior do sensor e o eixo y definido pela regra da mão direita. A origem do sistema de coordenadas {n} está a uma distância d da origem do sistema {0}. A Figura 3.23 ilustra a relação entre esses dois sistemas de coordenadas. Como o sensor gira em torno do eixo x do sistema de coordenadas {0}, a orientação do sistema de coordenadas {n} é alterada à medida que o valor de θ muda, gerando assim um novo sistema de coordenadas. O problema consiste, portanto,

em representar um ponto em um sistema de coordenadas genérico {n} em relação ao sistema de coordenadas de referência {0}.

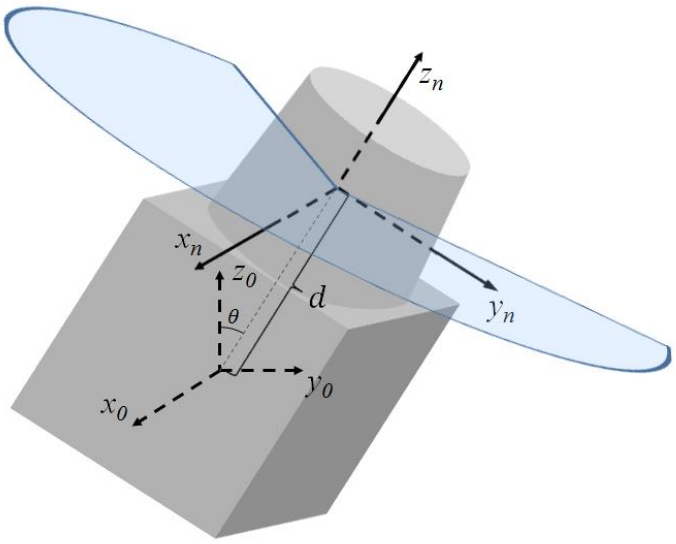


Figura 3.23 - Definição dos sistemas de coordenadas {n} e {0}
Fonte: Autoria própria

Os movimentos rígidos do sistema de coordenadas {n} em relação ao sistema {0} podem ser definidos como uma translação no sentido positivo do eixo z, seguida de uma rotação em torno do eixo x do sistema de coordenadas fixo. A partir das matrizes de transformação homogênea básicas, definidas em (3.19-21) e seguindo a regra de composição de transformadas, obtém-se a matriz H_n^0 que transforma as coordenadas do sistema {n} para o sistema {0}

$$H_n^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & sen\theta & 0 \\ 0 & -sen\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & sen\theta & dsen\theta \\ 0 & -\sin\theta & \cos\theta & d\cos\theta \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.24)$$

As coordenadas *(x, y)* de um ponto P em um plano varrido pelo sensor, podem ser obtidas da Figura 3.22 utilizando relações trigonométricas. Sendo ρ a distância medida pelo sensor quando o feixe está em ângulo β, as coordenadas do ponto P são dadas por:

$$\begin{cases} x = \rho cos\beta \\ y = \rho sen\beta \end{cases} \tag{3.25}$$

Portanto, aplicando a matriz de transformação homogênea obtida em (3.24), temos que as coordenadas *(x, y, z)* do ponto P em relação ao sistema de coordenadas fixo são dadas por:

$$\begin{bmatrix} P^0 \\ 1 \end{bmatrix} = H_n^0 \begin{bmatrix} P^n \\ 1 \end{bmatrix} \tag{3.26}$$

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & sen\theta & dsen\theta \\ 0 & -sen\theta & \cos\theta & d\cos\theta \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \rho\cos\beta \\ \rho sen\beta \\ 0 \\ 1 \end{bmatrix} \tag{3.27}$$

$$\begin{cases} x = \rho cos\beta \\ y = \rho sen\beta cos\theta + dsen\theta \\ z = -\rho sen\beta sen\theta + dcos\theta \end{cases} \tag{3.28}$$

3.2.5. Ferramenta de Visualização

Em posse das coordenadas tridimensionais dos pontos que descrevem o ambiente, deseja-se representar estes pontos graficamente. Como o volume de pontos é considerável (418748 pontos na resolução máxima), necessita-se de uma ferramenta robusta, que permita uma interação prática e eficiente do usuário com o mapa gerado. Um método que atende bem a esta aplicação é a representação em *Point Cloud* (ou nuvem de pontos). Point Cloud é uma estrutura de dados utilizada para representar um conjunto de pontos multidimensionais, comumente utilizada para representação de dados tridimensionais. O conceito de representação de dados tridimensionais em uma *Point Cloud* pode ser implementado com auxilio da *Point Cloud Library,* que é apresentada no tópico a seguir.

3.2.5.1. Point Cloud Library

Point Cloud Library (PCL) é uma biblioteca *open source* que incorpora algoritmos para manipulação e processamento de *Point Clouds.* A biblioteca é desenvolvida em C++, podendo ser implementada e compilada nos principais sistemas operacionais, como Linux, MacOS,

Windows e Android/iOS. Além disso, é completamente gratuita, podendo ser utilizada comercialmente e para fins de pesquisa. A PCL é composta de diversos módulos de processamento 3D como filtragem, segmentação, estimativa de características, reconstrução de superfícies, entre outros.

Um eficiente módulo de visualização também é disponibilizado, tornando possível a rápida e intuitiva interação com modelo tridimensional gerado. Neste trabalho, foi utilizada a ferramenta de visualização PCD Viewer que faz parte da biblioteca PCL. A ferramenta PCD Viewer possibilita configuração de propriedades visuais, como cores, tamanho dos pontos, e opacidade. O formato de arquivo suportado por essa ferramenta é o PCD (Point Cloud Data). PCD é um arquivo de texto que contém um cabeçalho com propriedades e a lista de coordenadas dos pontos de uma *Point Cloud* e é utilizado como arquivo de entrada e saída dos algoritmos da biblioteca PCL. Um exemplo de arquivo PCD é apresentado na abaixo.

```
# .PCD v.7 - Point Cloud Data file format
VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 16
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0
POINTS 16
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
0.81915 0.32 0 4.2108e+06
0.97192 0.278 0 4.2108e+06
0.944 0.29474 0 4.2108e+06
0.98111 0.24247 0 4.2108e+06
0.93655 0.26143 0 4.2108e+06
0.91631 0.27442 0 4.2108e+06
0.81921 0.29315 0 4.2108e+06
0.90701 0.24109 0 4.2108e+06
0.83239 0.23398 0 4.2108e+06
0.99185 0.2116 0 4.2108e+06
0.89264 0.21174 0 4.2108e+06
0.85082 0.21212 0 4.2108e+06
0.81044 0.32222 0 4.2108e+06
0.74459 0.32192 0 4.2108e+06
```

4. Resultados

Esta seção apresenta os resultados obtidos dos testes realizados com o sistema de mapeamento. O sensor Hokuyo URG-04LX-UG01 é destinado a aplicações em interiores, portanto os testes se restringem a estes casos. Foram feitos testes variados para verificar o desempenho geral do sistema, seu funcionamento em diferentes ambientes e detectar possíveis pontos fracos no mapeamento.

4.1. Análises Preliminares

A Figura 2.1 apresenta um mapa gerado pelo sistema. Pode-se localizar facilmente no mapa o ponto focal, ou seja, o ponto em comum entre os planos detectados pelo sensor. Como descrito na seção 2, na configuração de *rolling*, a maior densidade de pontos se concentra na área diretamente frontal ao sensor, onde se localiza o ponto focal. Oposta a este ponto, é possível identificar no mapa a região de oclusão, ou seja, uma área sem presença de pontos. Essa área corresponde aos 120° que estão fora da região varrida pelo sensor. Como neste

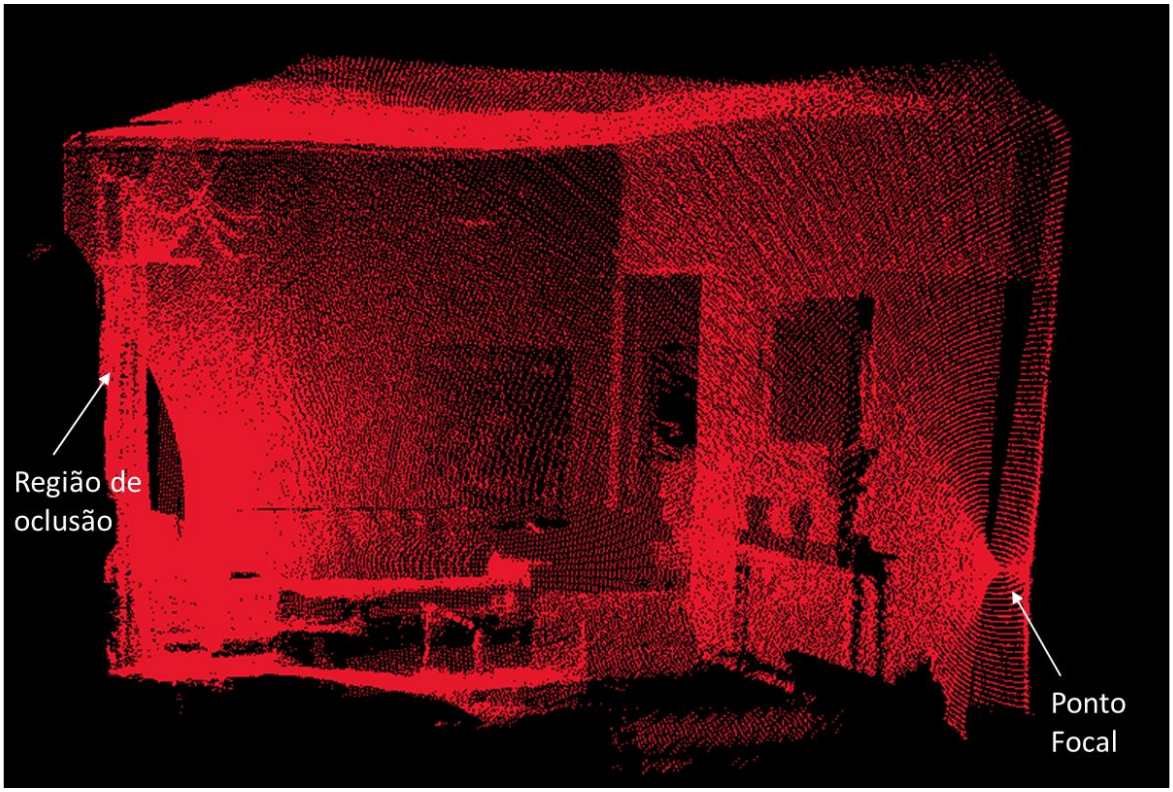


Figura 4.1 - Mapa de uma sala gerado pelo sistema
Fonte: Autoria própria

caso o sistema foi posicionado em um dos cantos da sala, a perda devido a região não coberta é mínima e se obtém um mapa consideravelmente representativo do ambiente.

As dimensões da sala permitem que todas as superfícies e objetos sejam detectados, o que valida a aplicação do sistema em ambientes relativamente pequenos. Evidentemente, algumas áreas aparecem sombreadas, pois como a medição do sistema é estática, se existem objetos no caminho do feixe a uma determinada superfície, apenas os objetos a sua frente serão detectados causando uma sombra nesta superfície.

Também se pode observar um pequeno desalinhamento no mapa (próximo ao ponto focal na **Erro! Fonte de referência não encontrada.**). Verificou-se que o mesmo ocorre no encontro dos pontos relativos aos primeiros com os últimos planos detectados pelo sensor. Este problema pode ser causado pelas operações trigonométricas realizadas com valores de distância medida (Equação 3.28). As funções de seno e cosseno de θ são calculadas com base na posição atual do eixo do motor (*Goal Position*). Neste caso, para que o encontro dos planos inicial e final estivesse perfeitamente alinhado, os valores de senθ e cosθ para essas posições deveriam ser rigorosamente os mesmos, o que não ocorre na prática. O fato do eixo de rotação estar deslocado do plano de varredura também é uma fonte de erro na geração do mapa. Mesmo que o efeito deste deslocamento seja corrigido através das transformações homogêneas, sempre haverá uma pequena área que não será mapeada. O processo de compensação deste erro remove esta pequena área do mapa causando um aparente descasamento na união dos planos inicial e final.

Outra peculiaridade observada é que objetos feitos de alguns tipos de materiais foram mal detectados ou não foram detectados de maneira alguma. Estes casos incluem superfícies translúcidas, espelhadas e metálicas. Na Figura 4.2, é apresentada uma comparação entre o mapa gerado e o ambiente real. Nota-se, por exemplo, que a superfície do espelho na parede ficou totalmente sem pontos. Esta é, de fato, uma deficiência do método LIDAR, como explica Tarini et al. (2003). Como o laser é refletido principalmente na direção especular da superfície, pouca ou nenhuma luz é refletida de volta ao sensor.

Um ponto interessante observado nos testes foi que o mapeamento do ambiente iluminado produziu exatamente o mesmo resultado do mapeamento do ambiente totalmente sem iluminação. Isto confirma o fato do mapeamento por LIDAR ser um método ativo, ou seja, independente de iluminação exterior. Esta característica possibilitaria, por exemplo, que o

sistema fosse integrado a um robô para que este pudesse se localizar em um ambiente não iluminado.

Figura 4.2 - Comparação da Point Cloud obtida com o ambiente real
Fonte: Autoria própria

4.2. Análise de Desempenho do Sistema

Um fator importante é a determinação de parâmetros como resolução e tempo de mapeamento (tempo necessário para realizar um *scan* completo). Foram realizados testes com a resolução máxima, metade, um quarto e um oitavo da resolução máxima, conforme indica a Tabela 4.1.

Tabela 4.1 – Comparação do tempo de mapeamento em função da resolução

	Resolução Angular (Θ)	Resolução Pontos	Total de Pontos	Tempo de Mapeamento
Máxima	0,29	614 x 682	418748	134 seg
1/2	0,58	307 x 682	209374	68 seg
1/4	1,16	154 x 682	105028	35 seg
1/8	2,33	77 x 682	52514	17 seg

Observa-se na tabela que a relação do número de pontos da Point Cloud e o tempo necessário para mapeamento é aproximadamente linear. A Figura 4.3 apresenta a comparação entre scans com resoluções diferentes, sendo os de maior resolução estão mais ao topo, seguindo a sequencia da Tabela 4.1. A frente de uma sala de aula foi mapeada e pode-se observar que não há tanta diferença entre os resultados dos scans com máxima resolução e metada de resolução máxima. Portanto, dependendo da aplicação, pode ser feito um mapeamento com resolução mais baixa e ainda assim, atender a um requisito de definição.

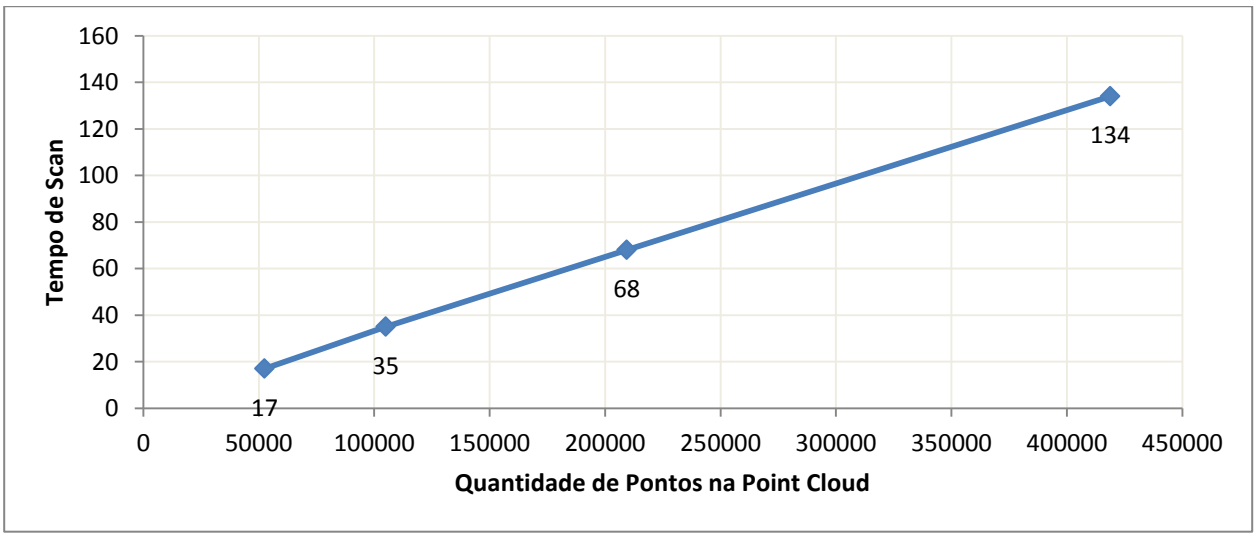


Gráfico 4.1 - Gráfico do tempo de scan em função da quantidade de pontos na Point Cloud

Figura 4.3 - Comparação de scans com resoluções diferentes
Fonte: Autoria própria

5. Conclusão

Este trabalho apresentou o projeto e construção de um sistema de mapeamento tridimensional de ambientes. Em geral, o sistema desenvolvido apresentou bons resultados. Pôde-se observar que o sistema teve melhor desempenho em ambientes pequenos, pois nestes todas as superfícies estavam ao alcance do feixe do sensor.

Apesar das restrições de alguns tipos materiais, grande parte dos objetos presentes no ambiente real podem ser identificados no mapa gerado, o que torna o sistema adequado para aplicações de detecção e reconhecimento de objetos, o que é muito comum em robótica.

Verificou-se também que, mesmo não operando na resolução máxima, o sistema produz mapas com considerável definição; o que significa um ganho no tempo necessário para o mapeamento. Na aplicação de SLAM, por exemplo, quando o intuito é gerar o mapa com o qual o robô navegará o ambiente, uma alta definição não é primordial. Nesse caso, é mais importante que o mapeamento seja feito rapidamente, pois se trata de uma aplicação dinâmica. Portanto, o sistema poderia operar com uma resolução mais baixa, mesmo assim possibilitando a implementação desejada.

5.1. Sugestões para trabalhos futuros

Um próximo passo seria integrar o sistema de mapeamento tridimensional a um robô autônomo. Isso pode ser feito a partir da plataforma ROS (*Robot Operating System)*, que possui módulos prontos para esse tipo de integração. Desta forma, poderiam ser implementadas aplicações como SLAM, detecção e desvio de obstáculo, entre outras.

Uma melhoria que pode ser feita neste sistema é a correção no eixo de rotação do sensor. A configuração na qual o sensor gira em torno de seu centro óptico apresentaria melhores resultados, minimizando os erros de desalinhamento na Point Cloud.

Referências

AYDAR, U.; AKYOL, O.; DURAN, Z. A low-cost laser scanning system design. In: **XXIIIth International CIPA Symposium, Prague**. 2011.

Evan S. Cameron; Ronald P. Szumski; James K. West. **Lidar scanning system.** US nº US5006721 A, 23 mar. 1990, 09 abr. 1991..

CHEN, S. et al. Active Vision Sensors. In: CHEN, S. et al. **Active sensor planning for multiview vision tasks**: Springer Berlin Heidelberg, 2008. p.11-38

FRANÇA, J.; GAZZIRO, M. A 3D scanning system based on laser triangulation and variable field of view. **IEEE International Conference on Image Processing, ICIP 2005**, v.1 p. 3–6, 2005.

GOLDBERG, S. Stereo vision and rover navigation software for planetary exploration. **Aerospace Conference Proceedings, 2002. IEEE ,** vol.5, p. 5-2025,5-2036 , 2002.

HÄHNEL, D.; MONTEMERLO, M. A system for volumetric robotic mapping of underground mines. **Submitted for publication**, p. 1–7, 2002.

KEMENY, J.; TURNER, K. **Ground-based lidar: rock slope mapping and assessment**. Lakewood, CO. Disponível em: <http://www.cflhd.gov/programs/techDevelopment/geotech/LiDAR/documents/01_ground_based_lidar_entire_document.pdf>. Acesso em: 5 nov. 2014.

LAING, R.; SCOTT, J. 3D high-definition scanning: Data recording and virtual modelling of the built heritage. **Journal of Building Appraisal**, v. 6, n. 3-4, p. 201–211, 2011.

MATTAUSCH, O. et al. Object detection and classification from large-scale cluttered indoor scans. **Computer Graphics Forum**, v. 33, n. 2, p. 11–21, 1 maio 2014.

MORALES, J. et al. Design and Development of a Fast and Precise Low-Cost 3D Laser Rangefinder. In: **Mechatronics (ICM), 2011 IEEE International Conference on**. IEEE, 2011. p. 621-626, 2011.

OKUBO, Y.; YE, C.; BORENSTEIN, J. Characterization of the Hokuyo URG-04LX laser rangefinder for mobile robot obstacle negotiation. In:**SPIE Defense, Security, and Sensing**. International Society for Optics and Photonics, 2009. p. 733212-733212-10. 2009.

ORIENTAL MOTOR. **Technical Reference**, 2012. Disponível em: < http://www.orientalmotor.com/products/pdfs/2012-2013/G/usa_tech_calculation.pdf> Acesso em: 5 nov 2014

RIISGAARD, S.; BLAS, M. SLAM for Dummies. **A Tutorial Approach to Simultaneous Localization and Mapping**, v. 22, n. 1-127, p. 126, 2003.

SHEH, R.; JAMALI, N. A low-cost, compact, lightweight 3d range sensor. In: **Australian Conference on Robotics and Automation**. 2006.

TÂCHE, F. et al. Three-dimensional localization for the MagneBike inspection robot. **Journal of Field Robotics**, v. 28, n. 2, p. 180–203, 19 mar. 2011.

TARINI, M. et al. **3D acquisition of mirroring objects**. MPI Informatik, Bibliothek & Dokumentation, 2003.

THRUN, S.; BURGARD, W.; FOX, D. A probabilistic approach to concurrent mapping and localization for mobile robots. **Autonomous Robots**, v. 271, p. 253–271, 1998.

THRUN, S.; BURGARD, W.; FOX, D. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In:**Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on**. IEEE, 2000. p. 321-328, 2000

TURNER, E. et al. Fast , Automated , Scalable Generation of Textured 3D Models of Indoor Environments. p. 1–15, 2014.

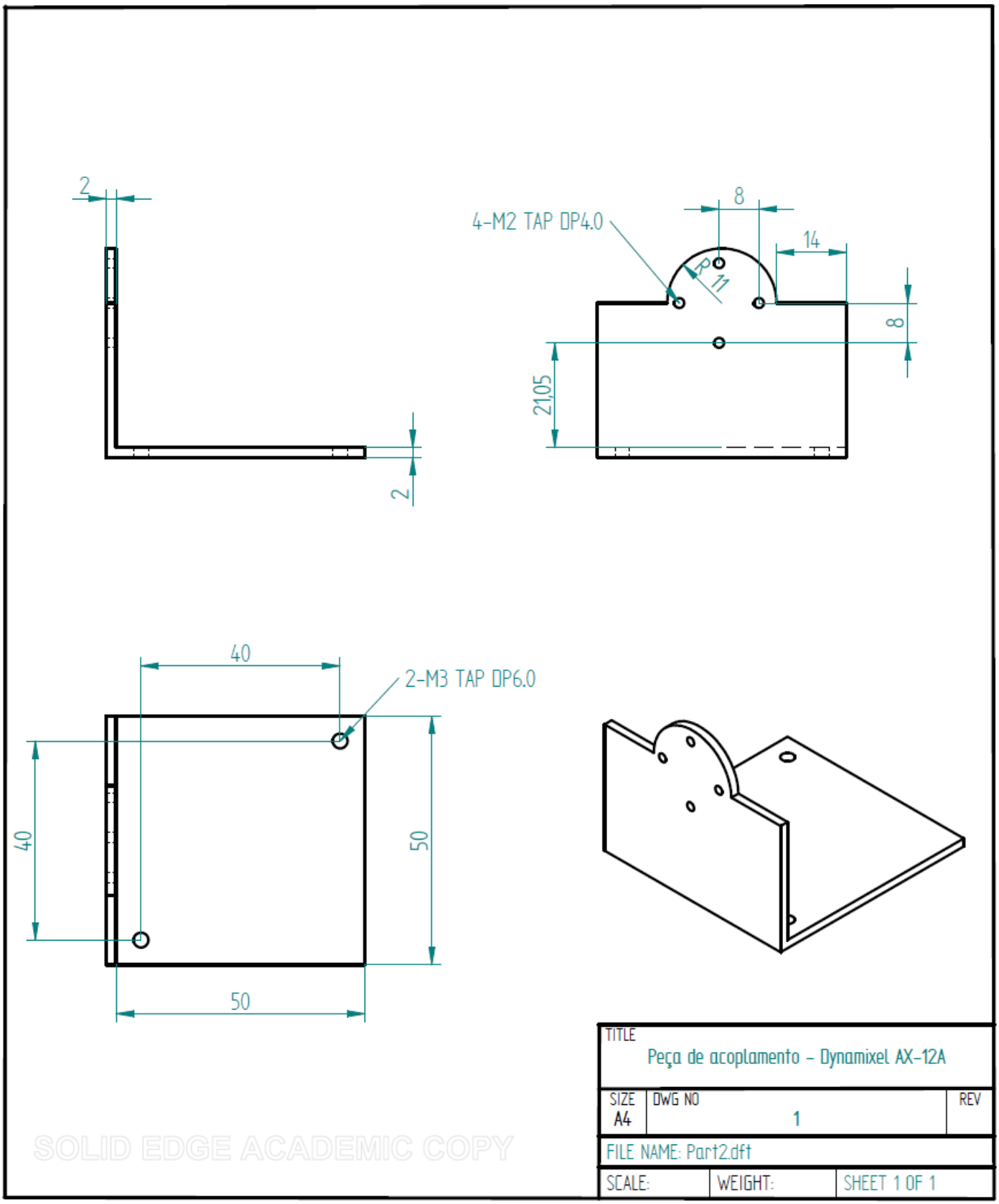
WATSON, A.; ULRICH, J. **Pidar: 3D Laser Range Finder**. Senior Design Documentation - University of Central Florida, 2014

WULF, O.; WAGNER, B. Fast 3D scanning methods for laser measurement systems. In: **International conference on control systems and computer science (CSCS14)**.p. 2-5, 2003

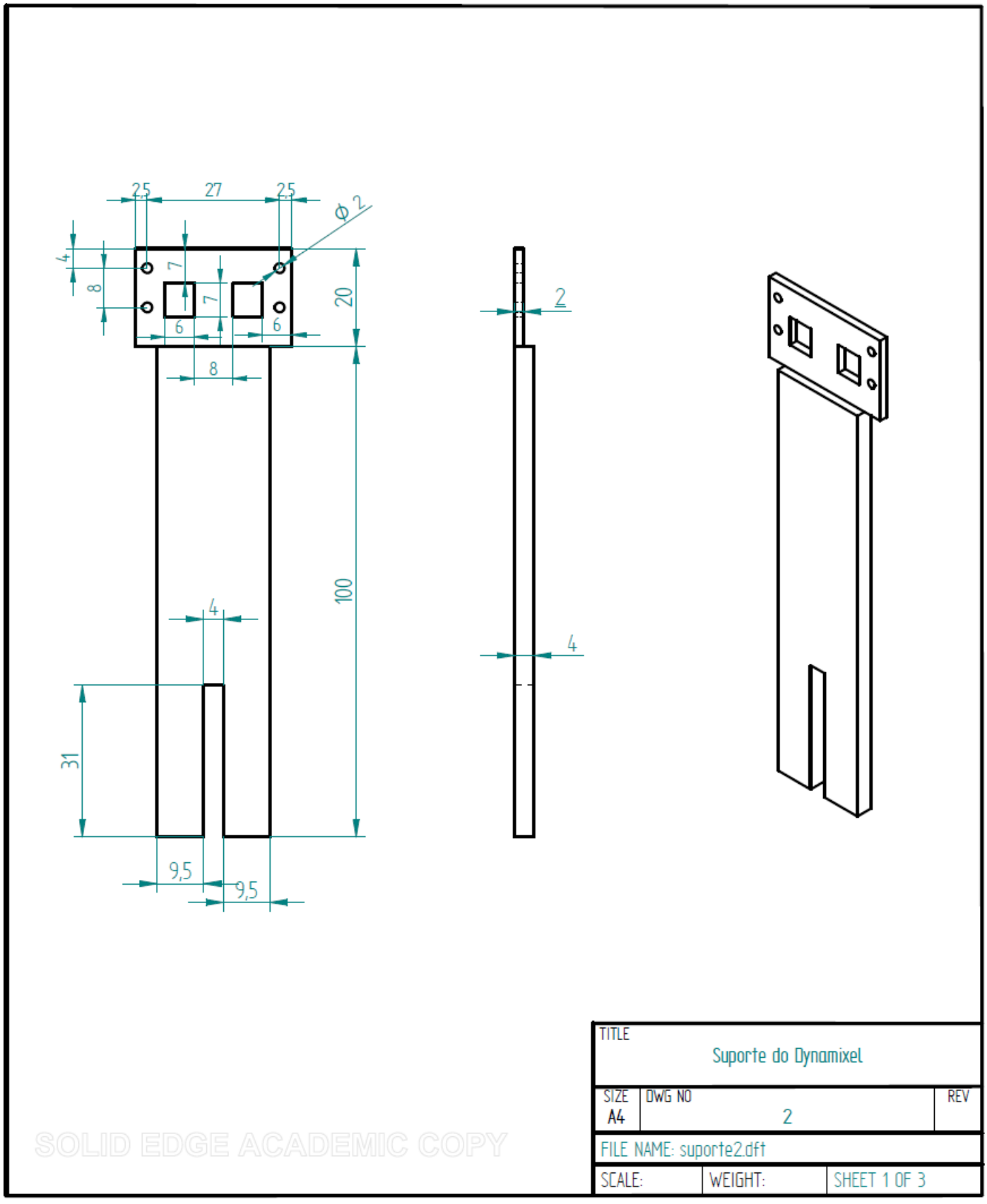
WUTKE, J. D. **Métodos para avaliação de um sistema laser scanner terrestre**. Dissertação (Mestrado) - Universidade Federal do Paraná, Curitiba. 2006.

YAMAUCHI, B.; LANGLEY, P.; SCHULTZ, A. MAGELLAN: An Integrated Adaptive Architecture for Mobile Robotics. v. 2, 1998.

Apêndice A - Diagrama da Peça de Acoplamento

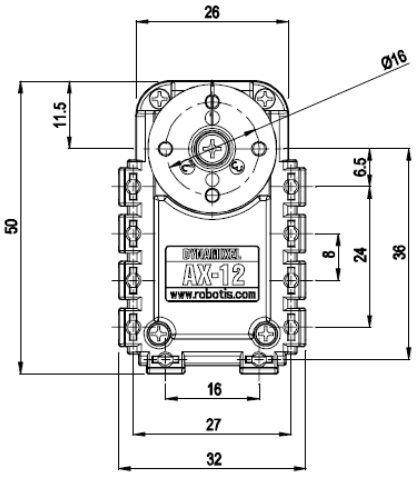


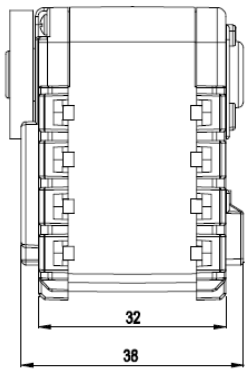
Apêndice B - Diagrama da Peça de Suporte



Apêndice C - Especificações do motor Dynamixel AX-12A

Parameter	Value
Weight	54.6 g
Dimension	32mm * 50mm * 40mm
Gear Reduction Ratio	254:1
Stall Torque @ Max Voltage	1.5 N.m (16.5 kg-cm)
No load speed (RPM)	59 (at 12V)
Nominal Operating Voltage	12 V
Stall Current Draw	1.5 A
Resolution	0.29°
Operating Angle	300 & Endless Turn
Geartrain Material	Engineering Plastic
Running Temperature	-5°C ~ +70°C
Onboard CPU	ATMega8 (ATMEGA8-16AU@16MHZ, B Bit)
Position Sensor	Potentiometer
Communication Protocol	Half duplex Asynchronous Serial Communication (8 bit, 1 stop, No Parity)
Link (Physical)	TTL
Communication Speed	7343 bps ~ 1 Mbps
Feedback	Position, Temperature, Load, Input Voltage, etc.



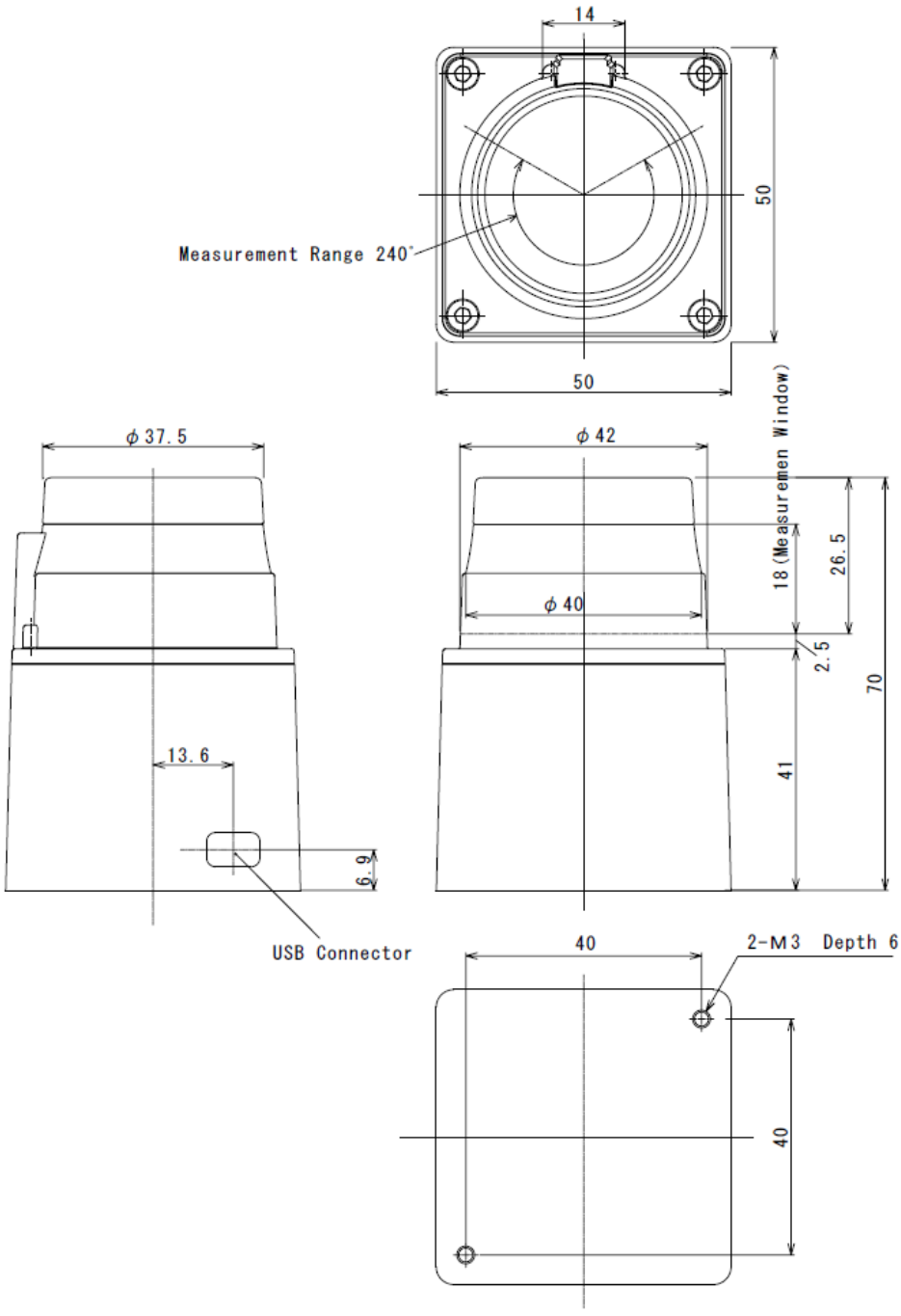


Apêndice D - Tabela de Controle do motor Dynamixel AX-12A

Area	Address (Hexadecimal)	Name	Description	Access	Initial Value (Hexadecimal)
EEPROM	0 (0X00)	Model Number(L)	Lowest byte of model number	R	12 (0X0C)
	1 (0X01)	Model Number(H)	Highest byte of model number	R	0 (0X00)
	2 (0X02)	Version of Firmware	Information on the version of firmware	R	-
	3 (0X03)	ID	ID of Dynamixel	RW	1 (0X01)
	4 (0X04)	Baud Rate	Baud Rate of Dynamixel	RW	1 (0X01)
	5 (0X05)	Return Delay Time	Return Delay Time	RW	250 (0XFA)
	6 (0X06)	CW Angle Limit(L)	Lowest byte of clockwise Angle Limit	RW	0 (0X00)
	7 (0X07)	CW Angle Limit(H)	Highest byte of clockwise Angle Limit	RW	0 (0X00)
	8 (0X08)	CCW Angle Limit(L)	Lowest byte of counterclockwise Angle Limit	RW	255 (0XFF)
	9 (0X09)	CCW Angle Limit(H)	Highest byte of counterclockwise Angle Limit	RW	3 (0X03)
	11 (0X0B)	the Highest Limit Temperature	Internal Limit Temperature	RW	70 (0X46)
	12 (0X0C)	the Lowest Limit Voltage	Lowest Limit Voltage	RW	60 (0X3C)
	13 (0X0D)	the Highest Limit Voltage	Highest Limit Voltage	RW	140 (0XBE)
	14 (0X0E)	Max Torque(L)	Lowest byte of Max. Torque	RW	255 (0XFF)
	15 (0X0F)	Max Torque(H)	Highest byte of Max. Torque	RW	3 (0X03)
	16 (0X10)	Status Return Level	Status Return Level	RW	2 (0X02)
	17 (0X11)	Alarm LED	LED for Alarm	RW	36(0x24)
	18 (0X12)	Alarm Shutdown	Shutdown for Alarm	RW	36(0x24)
RAM	24 (0X18)	Torque Enable	Torque On/Off	RW	0 (0X00)
	25 (0X19)	LED	LED On/Off	RW	0 (0X00)
	26 (0X1A)	CW Compliance Margin	CW Compliance margin	RW	1 (0X01)
	27 (0X1B)	CCW Compliance Margin	CCW Compliance margin	RW	1 (0X01)
	28 (0X1C)	CW Compliance Slope	CW Compliance slope	RW	32 (0X20)
	29 (0X1D)	CCW Compliance Slope	CCW Compliance slope	RW	32 (0X20)
	30 (0X1E)	Goal Position(L)	Lowest byte of Goal Position	RW	-
	31 (0X1F)	Goal Position(H)	Highest byte of Goal Position	RW	-
	32 (0X20)	Moving Speed(L)	Lowest byte of Moving Speed	RW	-
	33 (0X21)	Moving Speed(H)	Highest byte of Moving Speed	RW	-
	34 (0X22)	Torque Limit(L)	Lowest byte of Torque Limit	RW	ADD14
	35 (0X23)	Torque Limit(H)	Highest byte of Torque Limit	RW	ADD15
	36 (0X24)	Present Position(L)	Lowest byte of Current Position	R	-
	37 (0X25)	Present Position(H)	Highest byte of Current Position	R	-
	38 (0X26)	Present Speed(L)	Lowest byte of Current Speed	R	-
	39 (0X27)	Present Speed(H)	Highest byte of Current Speed	R	-
	40 (0X28)	Present Load(L)	Lowest byte of Current Load	R	-
	41 (0X29)	Present Load(H)	Highest byte of Current Load	R	-
	42 (0X2A)	Present Voltage	Current Voltage	R	-
	43 (0X2B)	Present Temperature	Current Temperature	R	-
	44 (0X2C)	Registered	Means if Instruction is registered	R	0 (0X00)
	46 (0X2E)	Moving	Means if there is any movement	R	0 (0X00)
	47 (0X2F)	Lock	Locking EEPROM	RW	0 (0X00)
	48 (0X30)	Punch(L)	Lowest byte of Punch	RW	32 (0X20)
	49 (0X31)	Punch(H)	Highest byte of Punch	RW	0 (0X00)

Apêndice E - Especificações do sensor Hokuyo URG-04LX-UG01

Parameter	Value
Light Source	Semiconductor laser diode (λ=785nm) Laser safety Class 1 (IEC60825-1)
Power Source	5V DC ±5% (USB buspower)
Current Comsumption	500mA or less (Rush current 800mA)
Detection Distance	20mm ~4000mm
Accuracy	Distance 20mm ~ 1000mm : ±30mm Distance 20mm ~ 4000mm : ±3% of measurement
Resolution	1 mm
Scan Angle	240°
Angular Resolution	0.36°
Scan Time	100msec/scan
Interface	USB Version 2.0 FS mode (12Mbps)
Ambient (Temperature/Humidity)	-10 ~ 50°C / 85% or less (without dew and frost)
Preservation Temperature	-25 ~75°C
Ambient Light Resistance	10000Lx or less
Impact Resistance	196 m/s^2, 10 times each in X, Y and Z direction
Protective Structure	Optics: IP64 Case: IP60
Insulation Resistance	10MΩ for DC 500Vmegger
Weight	Approx. 160 g
Case	Polycarbonate
External Dimension (WxDxH)	50mm*50mm*70mm


14
50
Measurement Range 240°
50
ϕ 37.5
ϕ 42
18 (Measuremen Window)
26.5
ϕ 40
2.5
70
41
13.6
6.9
USB Connector
40
2-M3 Depth 6
40

Apêndice F - Programa em MATLAB

Programa de Configuração da Comunicação Serial com o sensor Hokuyo (Fonte: http://www.mathworks.com/matlabcentral/fileexchange/36700-hokuyo-urg-04lx-lidar-driver-for-matlab)

```
% Setup Lidar
% % Configures Serial Communication and Updates Sensor Communication to
% SCIP2.0 Protocol.
% % Checks Version Information and switches on Laser.
% Author- Shikhar Shrestha, IIT Bhubaneswar
lidar=serial('COM4','baudrate',115200);
set(lidar,'Timeout',0.1);
set(lidar,'InputBufferSize',40000);
set(lidar,'Terminator','CR');
fopen(lidar);
pause(0.1);
fprintf(lidar,'SCIP2.0');
pause(0.1);
fscanf(lidar);
fprintf(lidar,'VV');
pause(0.1);
fscanf(lidar)
fprintf(lidar,'BM');
pause(0.1);
fscanf(lidar)
```

Programa Geral (incluindo configuração da comunicação serial do motor Dynamixel)

```
% Initialization - Check if USB2Dynamixel is connected
loadlibrary('dynamixel', 'dynamixel.h');
libfunctions('dynamixel');
DEFAULT_PORTNUM = 3;
DEFAULT_BAUDNUM = 1;

CONNECTION = calllib('dynamixel', 'dxl_initialize', DEFAULT_PORTNUM,
DEFAULT_BAUDNUM);

if CONNECTION == 0
   disp('Failed to open USB2Dynamixel')
end

if CONNECTION == 1
   disp('Succeed to open USB2Dynamixel!')
   P_GOAL_POSITION = 30;

   id = 1;
```

```
    init_pos = 207;
    final_pos = 820;
    frames = final_pos - init_pos + 1;

    scan_matrix = zeros(frames, 682);
    theta_array = zeros(frames,1);
    point_cloud = zeros(frames*682,3);

    D = 25; %Distancia entre o feixe do laser e o eixo do motor

    h = waitbar(0,'Scanning Environment. Please wait...');

    for j=1:frames
        waitbar(j/frames)
        GoalPos = j+(init_pos - 1);
        calllib('dynamixel','dxl_write_word',id,P_GOAL_POSITION,GoalPos);
        theta_array(j,1) = deg2rad(goalpos2deg(GoalPos));
        scan_matrix(j,:) = LidarScan(lidar);
    end

end
close(h)

calllib('dynamixel','dxl_terminate');

unloadlibrary('dynamixel');

% Point Cloud Generation

h2 = waitbar(0,'Generating Point Cloud... Almost Done!');

for k = 1:length(theta_array)
     waitbar(k/frames)
     theta = theta_array(k,1);
     for i = 1:size(scan_matrix,2)
            beta = deg2rad(laserpos2deg(i));
            P = scan_matrix(k,i);
            x = P*cos(beta);
            y = P*sin(beta)*cos(theta) + D*sin(theta);
            z = -P*sin(beta)*sin(theta) + D*cos(theta);

            act_scan(i,:)=[x y z];
     end


    index1 = 682*(k-1)+1;
    indexN = 682*k;
    point_cloud([index1:indexN],:) = act_scan;
end

close(h2)

savepcd('salad12b.pcd', point_cloud')
```