UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTOS ACADÊMICOS DE ELETRÔNICA E MECÂNICA
CURSO SUPERIOR DE TECNOLOGIA EM MECATRÔNICA INDUSTRIAL

ARILDO DIRCEU CORDEIRO JUNIOR

SISTEMA DE VISÃO ROBÓTICA APLICADO A TAREFAS DE MANIPULAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2013

ARILDO DIRCEU CORDEIRO JUNIOR

SISTEMA DE VISÃO ROBÓTICA APLICADO A TAREFAS DE MANIPULAÇÃO

Trabalho de Conclusão de Curso de Graduação, apresentado ao Curso Superior de Tecnologia em Mecatrônica Industrial, dos Departamentos Acadêmicos de Eletrônica e Mecânica, da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Técnologo.
Orientador: Marcelo Victor Wust Zibetti, Dr. Eng.
Co-Orientador: Luiz Carlos de Abreu Rodrigues, Dr. Eng.

CURITIBA
2013

TERMO DE APROVAÇÃO

ARILDO DIRCEU CORDEIRO JUNIOR

SISTEMA DE VISÃO ROBÓTICA APLICADO A TAREFAS DE MANIPULAÇÃO

Este trabalho de conclusão de curso foi apresentado no dia 26 de setembro de 2013, como requisito parcial para obtenção do título de Tecnólogo em Mecatrônica Industrial, outorgado pela Universidade Tecnológica Federal do Paraná. O aluno foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

_____
Prof. Dr. Milton Luiz Polli
Coordenador de Curso
Departamento Acadêmico de Mecânica

_____
Prof. Esp.Sérgio Moribe
Responsável pela Atividade de Trabalho de Conclusão de Curso
Departamento Acadêmico de Eletrônica

BANCA EXAMINADORA

_____
Prof. Dr. Hugo Vieira Neto
UTFPR

_____
Prof. Dr. Luís Paulo Laus
UTFPR

_____
Prof. Dr. Marcelo Victor Wust Zibetti
Orientador - UTFPR

"A Folha de Aprovação assinada encontra-se na Coordenação do Curso"

AGRADECIMENTOS

Agradeço a todos os colegas de classe que me auxiliaram tirando dúvidas e fornecendo sugestões durante o desenvolvimento deste projeto. Ao meu orientador, Prof. Dr. Zibetti, que me auxiliou durante toda esta jornada, direcionando conteúdos e transmitindo seus conhecimentos com seriedade e paciência. Aos Prof. Luís Paulo Laus e Prof. Luiz Carlos que me ensinaram conceitos fundamentais em robótica, além de aprimorarem o meu projeto com observações importantíssimas.

Uma dedicação especial ao meu pai Arildo, que me auxiliou durante toda a conceituação, desenvolvimento e conclusão do trabalho. Como sempre um pai maravilhoso, que sempre está disponível para o que for preciso, me dando força e motivação, estando ao meu lado nos momentos mais difíceis. Um pai que acredita realmente que eu não tenho limites e posso realizar todos os meus sonhos. É a maior referência e exemplo que tenho em minha vida.

Agradeço de maneira especial aos meus sócios que com muita paciência me acompanharam do início ao fim do trabalho, compreendendo minhas ausências na empresa e me auxiliando no que puderam.

Por fim, agradeço a todos meus familiares, em especial minha mãe Eliane e minha irmã Licyane por sempre me motivarem e acreditarem em mim durante a realização deste trabalho.

RESUMO

CORDEIRO JUNIOR, Arildo D. **Sistema de visão robótica aplicado a tarefas de manipulação.** 2013. 124 f. Trabalho de Conclusão de Curso (Tecnologia em Mecatrônica Industrial) – Departamentos Acadêmicos de Eletrônica (DAELN) e Mecânica (DAMEC), Universidade Tecnológica Federal do Paraná. Curitiba, 2013.

Este trabalho apresenta um sistema de visão robótica aplicado a tarefas de manipulação de peças. O sistema desenvolvido é capaz de reconhecer, localizar e manipular uma única peça localizada em uma posição inicialmente desconhecida pelo sistema. Foi criado um sistema composto de um robô caracterizado por um braço robótico que manipula a peça, uma câmera para capturar as imagens da peça localizada no seu plano de visão, o qual é materializado pela superfície de uma mesa confeccionada para esse fim e um computador para processar as imagens adquiridas pela câmera e enviar instruções para o robô. Para tanto, o projeto faz uso dos programas *Vision Builder* e *LabVIEW*, ambos da *National Instruments*. O sistema oferece flexibilidade com relação ao posicionamento do robô e da mesa, não há necessidade de ambos estarem alinhados ou serem paralelos, sendo que a peça pode estar em qualquer posição dentro do alcance do robô e do campo de visão da câmera. Foram realizados ensaios de laboratório para testar o funcionamento adequado do sistema de visão robótica. Os ensaios foram feitos a partir de uma calibração inicial, colocando-se a peça em duas posições pré-estabelecidas e após finalizada a calibração, o sistema visualiza e manipula a peça. Para realizar essas operações, foram utilizados conceitos de topografia e geodésia, devido ao sistema de visão estar em um sistema de coordenadas diferentes (originado pela câmera) do sistema de coordenadas do robô e também pela variação no posicionamento do robô e da mesa. Para a compatibilização dos sistemas de coordenadas foram necessários os conceitos citados para fazer rotação e translação, bem como para compensar os erros das medidas efetuadas. Portanto, este trabalho poderá ter aplicações importantes na indústria automatizada, considerando a sua flexibilidade, uma vez que o robô deixará de agir apenas por instruções recebidas manualmente, mas também através de percepções de mudanças no seu campo de alcance através da visão robótica.

Palavras-chave: Visão robótica. Processamento de imagens. Robô industrial. Lógicas de programação. Análise de dados. Rotação e translação de sistemas e manipulação de peças.

ABSTRACT

CORDEIRO JUNIOR, Arildo D. **Robotic vision system applied to manipulation tasks.** 2013. 124 p. Final Year Project (Industrial Mechatronics Technology) – Electronics (DAELN) and Mechanics (DAMEC) Academic Departments, Universidade Tecnológica Federal do Paraná. Curitiba, 2013.

This work presents a robotic vision system applied to part manipulation tasks. The system developed is capable of recognizing, locating and manipulating a single part located in a position initially unknown to it. A system was created, composed of a robot characterized by a robotic arm that manipulates the part, a camera to capture images of the part in its visual field, which is delimited by the surface of a table built for this purpose, and a computer to process the images captured by the camera and send instructions to the robot. Therefore, the project makes use of the Vision Builder and LabVIEW softwares, both developed by National Instruments. The system offers flexibility in relation to the position of the robot and the table - there's no need of both being aligned or parallel, as the part can be placed anywhere inside the reach of the robot and the visual field of the camera. Lab tests were made in order to test the proper operation of the robotic vision system. The tests were performed from an initial calibration, where the part is placed in two pre-established positions and after that point, the robot visualizes and manipulates the part. In order to execute these operations, topography and geodesy concepts were used, due to the vision system being in a different coordinate system than the one originated by the camera and the variation on the position of both the robot and the table. To assure the compatibility of the coordinate systems, the cited concepts were required, as to make rotations and translations, as well as to compensate errors in the measurements acquired. Therefore, this project can have important applications in the automation industry, considering its flexibility, since the robot will not only act based on manually input instructions, but also on perceptions of change in its field of reach through its robotic vision.

Keywords: Robotic vision. Image processing. Industrial robot. Programming logic. Data analysis. Systems rotation and translation and part manipulation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CCD	Dispositivo de carga acoplada - *Charge Coupled Device*
CLP	Controlador Lógico Programável
CPU	Unidade Central de Processamento - *Central Processing Unit*
DAELN	Departamento Acadêmico de Eletrônica
DAMEC	Departamento Acadêmico de Mecânica
FFT	Transformação Rápida de Fourier - *Fast Fourier Transform*
FMS	Sistemas Flexíveis de Manufatura – *Flexible Manufacturing Systems*
GB	*Gigabyte*
GPS	Sistema de Posicionamento Global - *Global Positioning System*
HD	Alta Definição - *High Definition*
MB	*Megabyte*
PC	Computador Pessoal - *Personal Computer*
SCARA	Braço Robótico Seletivo de Montagem - *Selective Compliant Articulated Robot Arm*
RAM	Memória de Acesso Aleatório - *Random Access Memory*
RIA	Instituto de Robótica da América - *Robotics Institute of America*
USB	Barramento Serial Universal - *Universal Serial Bus*
UTFPR	Universidade Tecnológica Federal do Paraná

SUMÁRIO

1. INTRODUÇÃO

O aspecto atual da indústria demonstra que há uma grande necessidade de aprimorar continuamente todas as variáveis e parâmetros que envolvem um processo industrial. Por exemplo, aumentar a produtividade, melhorar a qualidade dos processos, diminuir custos com mão-de-obra especializada e desperdícios e cumprir prazos relativamente curtos.

Nesse contexto surge a automação industrial que, segundo Rosário (2005), integra três grandes áreas: a eletrônica, a mecânica e a informática. A eletrônica é responsável pelo *hardware* a ser utilizado pelo sistema, assim como estabelecer as melhores formas de interconectar os seus componentes. A mecânica contribui com máquinas e ferramentas que atuarão no processo. A informática atua fornecendo o *software* que, através de instruções, controla todo o sistema.

Dentre essas máquinas, o robô industrial surgiu com o objetivo de executar tarefas de caráter repetitivo de forma a aumentar a flexibilidade do sistema envolvido, as que necessitam de alta precisão e exatidão, ou ainda atuar em processos que envolvam ambientes agressivos e de alto risco para o ser humano. Exemplos de tarefas de alto risco são operações sob alta pressão, como operações no fundo do mar, ou em alta temperatura.

Em 1921, o dramaturgo Karel Capek, deu origem a palavra Robot provinda da palavra tcheca *robota* que significa servo. Capek escreveu a peça teatral Rossum's Universal Robots (R.U.R.), na qual utilizou o conceito de que robôs, comandados pelos humanos, deveriam executar atividades que exigiam esforço físico, conforme Rosário (2005). Na década de 60, o primeiro robô foi oficialmente desenvolvido pela *Unimation*, companhia fundada por George Devol e Joseph F, e era um braço robótico capaz de realizar diferentes tarefas. Posteriormente, em 1981, o *Robotics Institute of America* (*RIA)* estabeleceu a definição oficial de um robô industrial: "Um robô industrial é um manipulador reprogramável, multifuncional, projetado para mover materiais, peças, ferramentas ou dispositivos especiais em movimentos programáveis variáveis para a realização de uma variedade de tarefas."

Tendo em vista esta definição, o robô industrial pode ser considerado uma máquina atuante em sistemas que necessitam de flexibilidade. Este contexto aplica-se aos sistemas flexíveis de manufatura (FMS) que são capazes de produzir peças ou produtos rapidamente, com alta precisão e qualidade, conforme Groover (2008).

Para que isto seja possível, o sistema deve ser capaz de se adaptar às mudanças instantâneas no planejamento da produção, de processar peças de maneira econômica, de agir automaticamente diante de possíveis problemas, como parada de máquina de forma que não comprometa o processo; e finalmente, de se adequar a diferentes modelos de peças e produzi-las de maneira rápida e eficiente.

O presente trabalho foi realizado e desenvolvido dentro do Sistema Flexível de Manufatura (sala A-007) da UTFPR campus Curitiba. Um dos componentes principais do sistema é o robô Mitsubishi RV-M1, responsável por manipular todas as peças que serão produzidas.

1.1 Justificativa

O robô Mitsubishi RV-M1, atualmente, é controlado por instruções programadas que o movem para posições/orientações espaciais pré-definidas. As instruções podem ser acionadas através da interação com o Controlador Lógico Programável (CLP) do FMS (Sistema Flexível de Manufatura) ou utilizando um computador. Com isso, o robô apenas executa comandos específicos que o direcionam para uma coordenada fixa.

Em geral, o acionamento do robô ocorre quando ele recebe o sinal de um sensor externo, que apenas identifica a presença de peças na posição de carga. Contudo, não se identificam eventos em torno do espaço de trabalho. Isso significa que uma mudança de posição e orientação das peças ou possíveis perturbações podem inviabilizar a operação. Assim, concluiu-se que um elemento com sensor de visão poderia auxiliar as tarefas de manipulação executadas pelo braço robótico. Tal elemento é conhecido como sistema de visão robótico.

1.2 Objetivos

1.2.1 Objetivo Geral

Desenvolver e aplicar um sistema de visão robótico capaz de reconhecer, localizar e manipular uma peça cilíndrica de altura pré-definida, produzida no FMS (Sistema Flexível de Manufatura). A peça pode estar localizada em qualquer ponto sobre uma área pré-determinada, a qual coincide com o campo visual do sistema de visão e com o alcance do robô industrial.

1.2.2 Objetivos Específicos

- **Definição do ambiente de trabalho:** estabelecer um campo de visão para o robô, definir qual tipo de iluminação que será utilizado;
- **Definição dos objetos de pesquisa:** definir qual peça ou objeto será utilizado na aplicação do sistema a ser desenvolvido, definir qual câmera será utilizada para captação de imagens;
- **Desenvolvimento do programa:** desenvolver um programa de processamento de imagens no *Vision Builder* que processe e interprete imagens originadas pelo sensor de visão (câmera) afim de reconhecer e localizar objetos cilíndricos dentro de uma determinada área de trabalho, também desenvolver um programa no ambiente de desenvolvimento do LabVIEW capaz de aplicar lógicas de programação para analisar os dados adquiridos no programa de processamento de imagens e gerar instruções para o robô manipular uma peça localizada em um ponto desconhecido pelo sistema;
- **Integração do sistema físico:** estabelecer comunicação entre o computador que processa as imagens e o robô industrial Mitsubish RV-M1, de forma a viabilizar a troca de informações entre ambos.
- **Realização de ensaios experimentais:** realizar ensaios em laboratório de forma a testar o funcionamento adequado do sistema de visão que se caracteriza pela integração de uma câmera, um computador e um robô. Os ensaios serão feitos a partir de uma calibração inicial, colocando-se a peça em duas posições pré-estabelecidas e após finalizada a calibração, o sistema visualiza e manipula a peça em qualquer posição dentro da área de trabalho do robô e do campo de visão da câmera.

1.3 Metodologia do trabalho

O desenvolvimento desse projeto pode ser dividido nas seguintes etapas:

1. Revisão bibliográfica: consistiu em realizar estudos e revisões sobre o funcionamento do robô, conceitos de programação, visão computacional e processamento de imagens, até obter todo o embasamento teórico necessário para desenvolver o trabalho.

2. Programação no *software Vision Builder*: foi desenvolvido um programa capaz de processar e interpretar imagens originadas pelo sensor de visão (câmera) a fim

de reconhecer e localizar objetos cilíndricos dentro de uma determinada área de trabalho;

3. Programação no *software LabVIEW*: foi desenvolvido um programa capaz de analisar os dados adquiridos no programa de processamento de imagens (*Vision Builder*) e aplicar algoritmos a fim de gerar instruções para o robô manipular uma peça localizada em um ponto desconhecido pelo sistema;

4. Teste do *software* no *Vision Builder*: foi analisado se a capacidade de processamento das imagens e as funcionalidades oferecidas pelo *software* desenvolvido no *Vision Builder* são suficientes para alcançar o objetivo de analisar, processar e interpretar imagens originadas pelo sensor de visão (câmara).

5. Adaptação do programa de comunicação: foi realizada a verificação detalhada dos padrões de comunicação utilizados pelo robô. Foi estudado um programa exemplo existente no *software LabVIEW* para comunicação PC – Serial, o qual foi adaptado para ser possível enviar instruções do computador diretamente para o robô.

6. Teste do *software* no *LabVIEW*: foi analisado que a capacidade de processamento e as funcionalidades oferecidas pelo *software* desenvolvido no *LabVIEW* são suficientes para alcançar o objetivo de analisar e processar os dados adquiridos no programa de processamento de imagens e aplicar algoritmos a fim de gerar instruções para a tarefa de manipulação estipulada para o robô.

7. Teste prático do sistema de visão: a tarefa de manipulação do robô foi testada em laboratório até alcançar o objetivo final. Tal objetivo consistiu na verificação constante dos sinais da câmera, das imagens geradas pelo *software* de visão e das informações enviadas em tempo real ao robô industrial.

2. FUNDAMENTAÇÃO TEÓRICA

Para que haja um entendimento maior do trabalho, conceitos básicos ligados diretamente ao seu desenvolvimento serão apresentados, tais quais: caracterização dos robôs industriais, noções de visão computacional e processamento de imagens, caracterização da comunicação da interface do robô e descrição do *software* LabVIEW.

2.1 O Robô

Conforme Rosário (2005), um robô industrial é constituído pelos seguintes elementos:

- **Manipulador:** é um braço robótico capaz de manipular objetos através do posicionamento de seu efetuador no espaço. O braço é formado por uma base fixa e pela junção de elos e juntas que atuam as operações do robô. Os elos são as partes rígidas do robô e as juntas são eixos que interligam dois elos, o que caracteriza uma articulação, similar ao braço humano. A disposição das juntas e elos de um braço manipulador estabelece um limite em sua movimentação, o qual determina um alcance máximo do braço que delimita o espaço de trabalho do robô. Em virtude das variadas configurações dos eixos ou juntas, o robô pode ser movimentado de diferentes maneiras, classificadas da seguinte maneira, conforme a Figura 1:
- **Robôs de coordenadas cartesianas:** 3 eixos com movimento em linha reta, seja horizontalmente ou verticalmente;
- **Robôs de coordenadas cilíndricas:** 1ª junta (ou eixo) com movimento de rotação, enquanto a 2ª e 3ª juntas (ou eixos), perpendiculares entre si, têm movimento retilíneo; o que forma um espaço de trabalho no formato de um cilindro;
- **Robôs de coordenadas esféricas:** os dois primeiros eixos são capazes de realizar um movimento rotacional, enquanto o 3º eixo realiza movimentos retilíneos. Com isso o espaço de trabalho possui o formato de uma esfera;
- **Robôs SCARA:** possuem duas juntas de revolução e uma deslizante. São ideais para operações que exijam precisão e velocidade;
- **Robôs Antropomorfos ou Articulados:** normalmente é constituído por seis eixos rotativos mecânicos com capacidade de movimentação rotativa. As movimentações são similares a um braço humano.

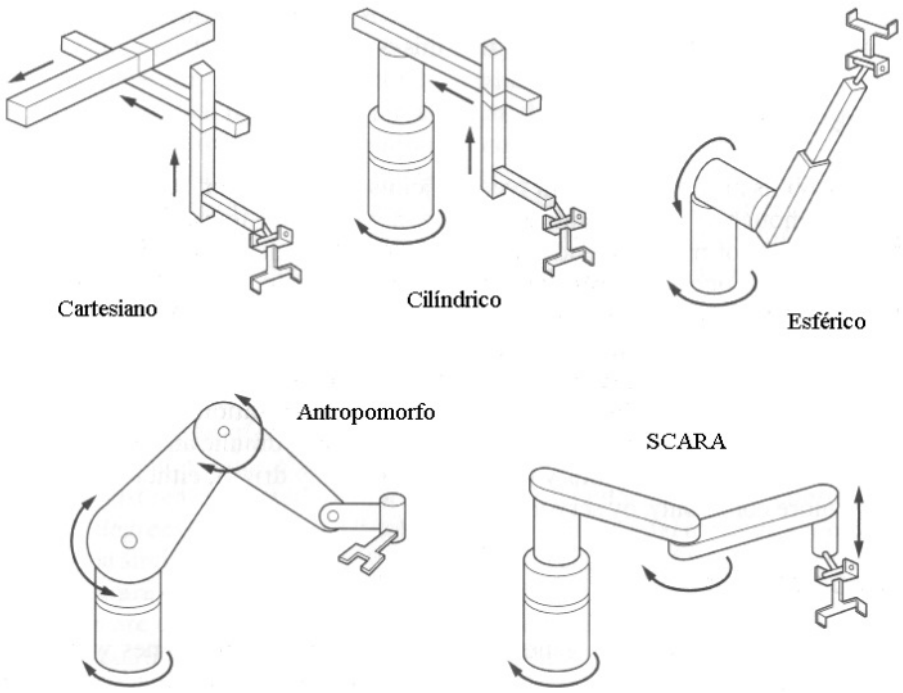


Figura 1: Tipos de Robô.
Fonte: GRASSI, (2005)

- **Efetuadores:** um efetuador é definido como a parte que está conectada à última junta do robô. É uma ferramenta, por exemplo uma garra, utilizada em tarefas de manipulação;
- **Sensores:** responsáveis por obter informações sobre o estado atual do robô e sobre o ambiente em que se encontra. Sensores internos ou proprioceptivos monitoram as articulações dos robôs, o que possibilita o posicionamento correto do manipulador no espaço. Sensores externos ou exteroceptivos podem ser utilizados para extrair informações do espaço de trabalho do robô. Assim é possível a implantação de soluções de segurança, como um sistema de visão, o qual é tema deste trabalho;
- **Atuadores:** são dispositivos que fornecem energia às partes atuantes do robô que originam a movimentação e posicionamento do mesmo. São classificados da seguinte maneira:
  - **Elétricos:** compostos normalmente por servo-motores;
  - **Hidráulicos**.
- **Unidade de controle:** módulo com configuração similar ao de um computador, constituído por processador e memória, que controla todo o movimento efetuado pelo manipulador. As informações originadas dos sensores internos são enviadas

ao controlador que efetua os cálculos necessários para a correta orientação do robô no espaço. Com isso, instruções são enviadas aos atuadores que efetivam a movimentação desejada.

Há parâmetros que determinam os limites e capacidades de um robô; dos quais os mais relevantes estão apresentados a seguir:

- **Precisão:** capacidade de realizar uma trajetória no espaço de trabalho sem ultrapassar um dado valor de erro;
- **Repetibilidade:** capacidade do robô, após ter se locomovido a outro ponto no espaço, de posicionar-se novamente num mesmo ponto pré-definido. Associa-se um limite ao erro de posicionamento num ponto de destino do efetuador;
- **Velocidade:** a rapidez com que o robô posiciona o seu efetuador é limitada pela velocidade máxima das juntas;
- **Capacidade de carga:** indica o quanto de carga é possível o robô suportar (incluindo o peso do efetuador/ferramental);
- **Número de eixos:** quantidade de juntas que determinam o número de graus de liberdade do robô;
- **Comunicação:** tipos de comunicação disponíveis, que possibilitam a entrada e saída de informações.

2.2 Robô modelo RV-M1

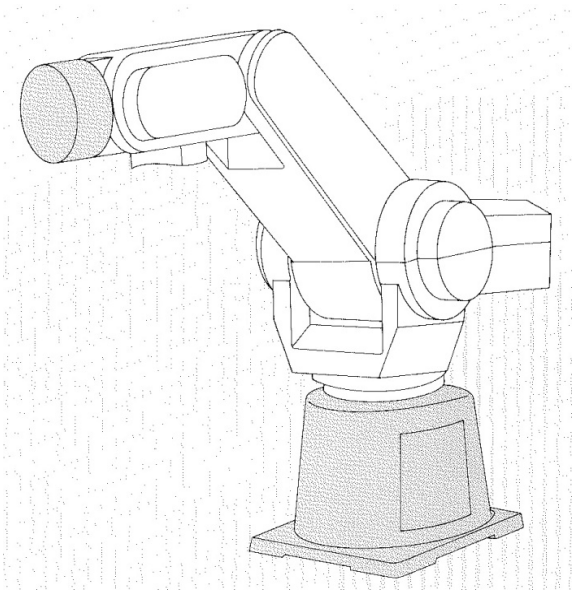


Figura 2: Robô RV-M1
Fonte: MITSUBISH, (1994)

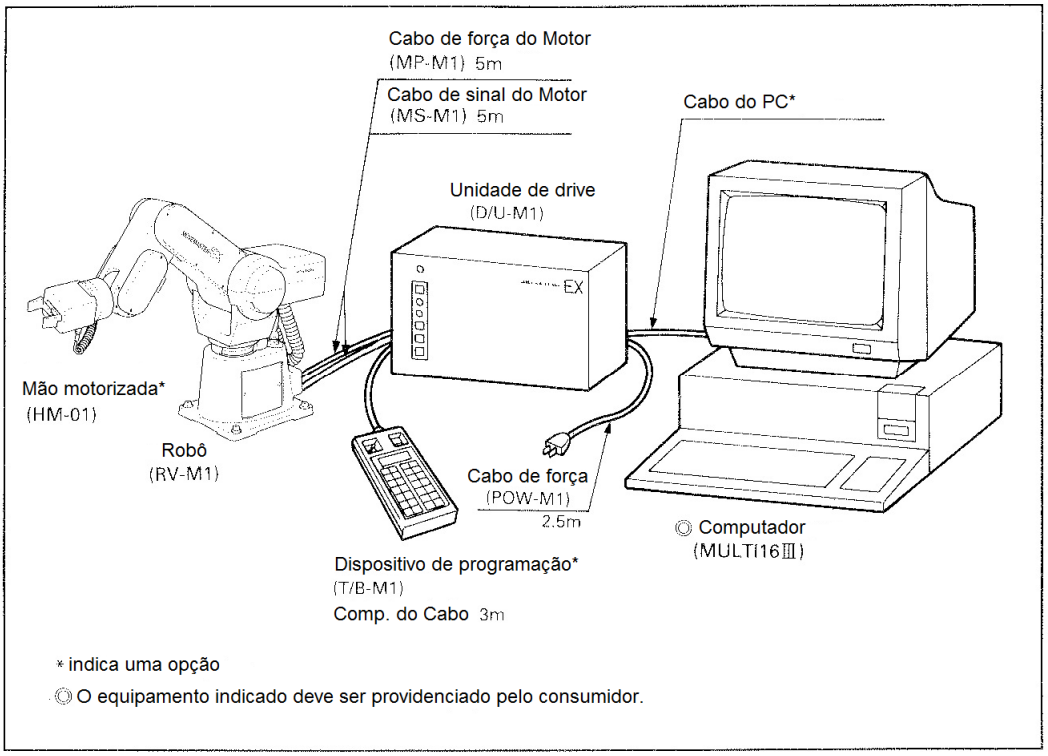


Figura 3: Conjunto de elementos do robô RV-M1
Fonte: Traduzido de MITSUBISH, (1994)

Este trabalho visa aplicar um sistema de visão de forma a auxiliar as tarefas realizadas pelo robô RV-M1 (Figura 2), marca Mitsubishi, no Sistema Flexível de Manufatura da UTFPR, situado na sala A-007. Este robô é constituído por um conjunto de elementos (Figura 3): o braço manipulador, a unidade de controle, o dispositivo de programação ou *teachingbox* (para de envio de instruções manualmente) e um computador (para execução de instruções pré-programadas e interface com o robô).

As especificações do robô são apresentadas na tabela dada a seguir:

Repetibilidade	Até 0,3mm
Carga admissível	Até 1,2kgf
Número de eixos	5
Velocidade	Até 1m/s
Comunicação	Canal serial (RS 232C) e paralela

Tabela 1: Especificações do robô RV-M1
Fonte: MITSUBISH, (1994)

2.3 Visão Robótica, Aquisição e Processamento de Imagens

2.3.1 Visão Robótica

Um sistema de visão é definido como um sistema computadorizado capaz de adquirir, processar e interpretar imagens correspondentes a cenas reais, conforme Marques Filho & Vieira Neto (1999). Um robô que possui um sistema de visão tem uma capacidade sensorial aumentada e, potencialmente, uma melhor interação com o ambiente que o circunda, como indicado por Foresti (2006). Essa condição pode permitir identificar eventos e responder a alterações operacionais como, por exemplo, um robô industrial em uma linha produção com a função de manipular peças que possuem variadas formas e que variam o seu posicionamento sobre uma determinada área.

2.3.2 Processamento de imagens

O procedimento de processar imagens, segundo Foresti (2006), pode ser dividido em etapas, ilustradas na Figura 4. Para assimilar os conceitos básicos de processamento de imagens, é usado como exemplo, o problema proposto por esse trabalho, de forma a facilitar o seu entendimento e expor os desafios e dificuldades a serem enfrentadas durante o seu desenvolvimento.

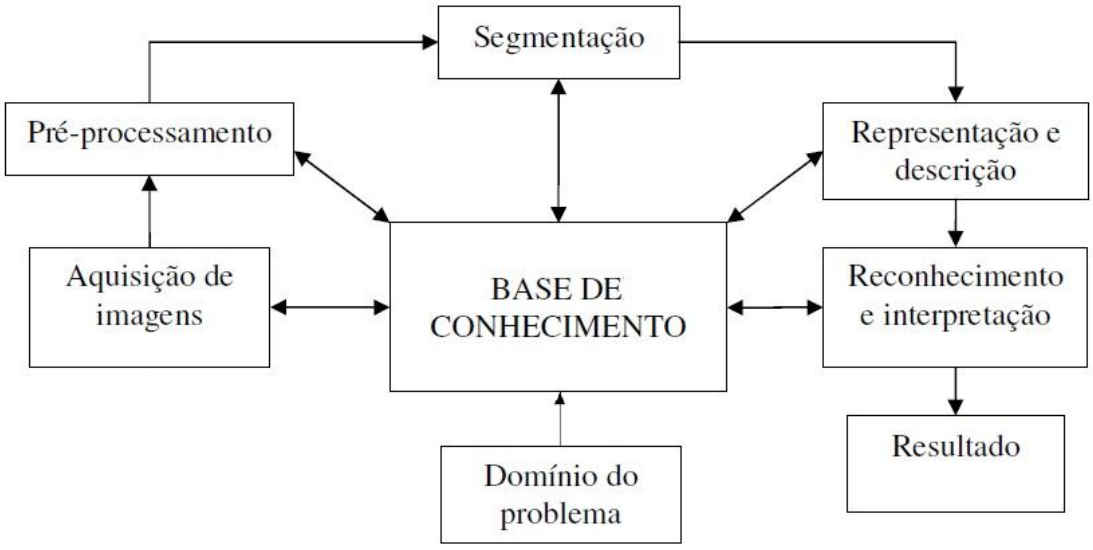


Figura 4: Etapas fundamentais em processamento de imagens.
Fonte: Adaptado de FORESTI, (2006)

- **Domínio do problema:** estabelecer o problema a ser solucionado, de tal maneira que se defina um objetivo a ser alcançado. O problema deste trabalho consiste em identificar e manipular uma peça cilíndrica, localizada em qualquer posição, sobre uma superfície plana limitada.

- **Aquisição de imagens:** para processar as imagens, primeiramente é necessário adquirir uma imagem digital. Para tanto, é necessário uma câmera capaz de captar as informações ópticas e traduzi-las em sinais elétricos que posteriormente serão digitalizados por um conversor analógico-digital, conforme Gonzalez & Woods (2000). Sendo assim, esta etapa produz na saída uma imagem digitalizada do espaço em que se situa a peça a ser manipulada.

- **Pré-processamento:** conforme Marques Filho & Vieira Neto (1999), esta etapa objetiva melhorar a qualidade das imagens adquiridas, de forma a diminuir a complexidade das etapas posteriores. A imagem proveniente da aquisição pode apresentar algumas irregularidades, como *pixels* ruidosos, tonalidades inadequadas, brilho ou contraste em níveis que dificultam a interpretação da cena, entre outras. Dentre as várias técnicas utilizadas para o pré-processamento de imagens podemos destacar: realce de contraste, remoção de ruído, isolamento de regiões de interesse e equalização de histograma de níveis de cinza. Portanto, a saída desta etapa é uma imagem com características mais adequadas para solucionar o problema. Neste exemplo, o resultado é uma imagem que destaca as formas e a localização da peça cilíndrica.

- **Segmentação:** conforme Grassi (2005), é em geral a etapa de maior complexidade em todo o processo, haja vista que o objetivo é separar o que é de interesse do restante da imagem. No caso de reconhecimento de objetos, o objetivo é reconhecer formas geométricas, área, dimensões, número de orifícios, ou outras características. Entretanto, para localizar objetos no espaço exibido na imagem, é necessário que existam pontos de referência para estimar a posição do objeto. Para reconhecer uma peça cilíndrica, esta etapa reconhece uma forma de círculo na imagem, desde que seja vista de

cima, e analisa, por exemplo, o diâmetro da peça. Contudo, para localizá-la, procura-se executar algoritmos que forneçam valores aproximados da localização do objeto na imagem. Para facilitar a localização, podem-se traçar linhas referenciais na superfície em que o objeto se situa, originando coordenadas que facilitam a execução desta operação.

- **Representação:** a representação contribui para que, posteriormente, os dados provenientes da segmentação sejam adequadamente processados no ambiente computacional.

- **Descrição:** objetiva extrair características da imagem, através de descritores, que propiciam informações suficientes para distinguir classes de objetos. Considerando que o objeto de interesse na imagem é uma peça cilíndrica, os descritores teriam o papel de analisar se o objeto possui contornos de natureza circular, assim como verificar se a peça não contém orifícios. Assim é diferenciada a peça de outras classes de objetos.

- **Reconhecimento e interpretação:** nesta última etapa, segundo Gonzales & Woods (2000), o reconhecimento atribui um rótulo a um objeto, através das informações fornecidas pelos descritores no passo anterior. A interpretação, analisa os objetos reconhecidos e atribui a eles um significado. No presente exemplo, os descritores viabilizam que a peça a ser manipulada possua o rótulo *cilíndrica*. Com isso, ao analizar os valores aproximados da localização do objeto, originados na etapa de segmentação, é interpretado que o objeto reconhecido é uma peça **cilíndrica** localizada na posição (x,y) do plano da imagem.

- **Base de Conhecimento:** é uma base que armazena o conhecimento do problema a ser resolvido. De acordo com Marques Filho & Vieira Neto (1999), todas as etapas do processamento da imagem são guiadas pela base de conhecimento, assim como a interação entre elas, representada através de setas duplas na Figura 4. Por exemplo, a etapa de representação e descrição recebe uma imagem sem objeto algum a ser manipulado. Uma provável responsável pela falha é a etapa de segmentação. Assim, uma realimentação

com esta etapa é estabelecida, de sorte que constantemente uma nova imagem é segmentada, até que se adéque ao conteúdo da base de conhecimento.

2.3.3 Aquisição de imagens

De acordo com Bovik (2009), a maioria das imagens digitais se origina de um tipo de radiação. Praticamente todos os tipos de radiação utilizados para geração de imagens, tais como raios-X e infravermelho, se situam em uma faixa do espectro eletromagnético. Os olhos humanos, por exemplo, são capazes de ver apenas um pequeno intervalo dessa faixa, conforme Figura 5. Bovik (2009) e Gonzalez & Woods (2000) explicam que o processo de formação da imagem, de maneira geral, ocorre quando alguma fonte emite um determinado tipo de radiação, depois interage com algum material para que posteriormente um dispositivo físico, sensível a uma banda do espectro de energia eletromagnética, perceba o nível de energia de forma a produzir proporcionalmente sinais elétricos de saída.

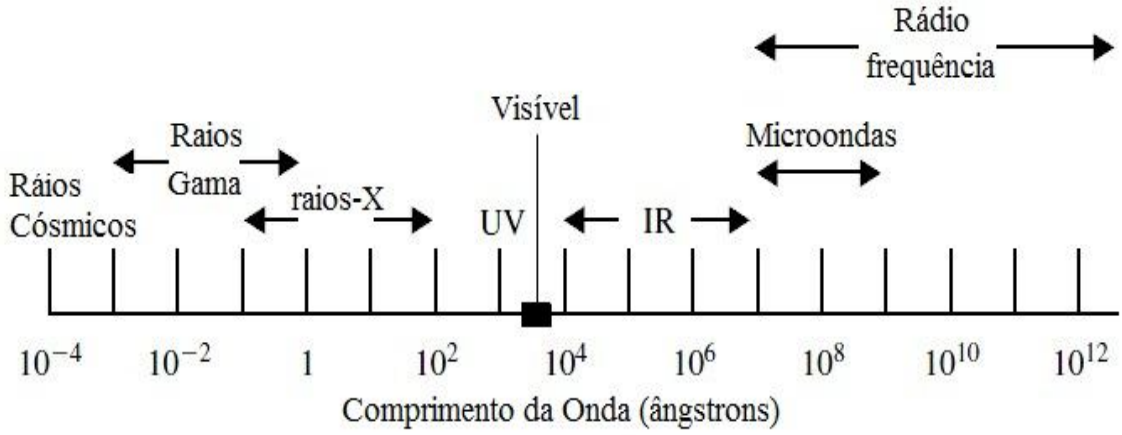


Figura 5: O espectro eletromagnético.
Fonte: Traduzido de BOVIK, (2009)

Dentre os dispositivos de aquisição de imagens existentes, uns dos mais utilizados são os dispositivos de carga acoplada – CCD (*Charge Coupled Device*). Conforme Foresti (2006), os CCDs são dispositivos constituídos por um grupamento de células semicondutoras capazes de armazenar carga elétrica proporcional à radiação luminosa incidente. A quantidade de carga armazenada traduz-se em valores analógicos que posteriormente originarão sinais que caracterizam a imagem captada pela câmera.

Os sinais de saída gerados pelos sensores de luz em geral possuem natureza analógica, ou seja, pertencem ao domínio contínuo. Porém, para que um sistema computacional reconheça adequadamente tais sinais, é necessário que os mesmos estejam no formato digital, ou seja, pertencentes ao domínio espaço/tempo discreto, conforme Bovik (2009). Assim, surgiu a necessidade de converter sinais analógicos em digitais, cuja função é executada por um digitalizador que enviará os dados digitais para um dispositivo externo realizar o processamento das imagens, por exemplo, um computador.

2.3.4 Técnicas de iluminação

Na inspeção de um objeto em um sistema de visão, é importante que haja um sistema de iluminação que o destaque, de modo que fatores negativos às imagens como sombras, baixo contraste ou reflexos sejam amenizados. Consequentemente, de acordo com Grassi (2005), a complexidade da análise feita pelo sistema é diminuída de maneira considerável, sendo assim, a qualidade de saída do processo de aquisição é suficiente para se executar as tarefas do processamento digital das imagens com um custo computacional minimizado.

Um sistema de iluminação para a visão robótica é constituído de uma fonte de luz que ilumina o espaço de trabalho do robô. Grande parte dos feixes luminosos deve incidir no objeto de estudo do sistema de visão, de forma que realce suas principais características visuais.

Conforme Pavim (2005), a técnica de iluminação direta possui a capacidade de realçar as características do objeto de interesse, já que a luz incide diretamente sobre o mesmo. Utilizando esta técnica neste trabalho, a forma geométrica do objeto é destacada, o que facilita a identificação do mesmo durante o processamento das imagens adquiridas da câmera. Com isto, concluiu-se que tal técnica é a mais adequada para este trabalho.

2.3.5 Pré-processamento

As imagens provenientes da etapa de aquisição geralmente possuem características que desfavorecem os processamentos posteriores, tais como sombras, excesso de brilho, contraste irregular, pequenas falhas nos *pixels*, etc. Portanto, a etapa de pré-processamento consiste em realçar as características do objeto de estudo e melhorar ao máximo a qualidade das imagens.

As técnicas de pré-processamento utilizadas neste trabalho utilizaram informações provindas do histograma da imagem capturada. O histograma é a representação gráfica da frequência de distribuição dos níveis de cinza de uma imagem.

Estão brevemente introduzidas abaixo, algumas das principais técnicas de pré-processamento digital de imagens:

I. **Manipulação de contraste e brilho:** ao avaliar as informações contidas no histograma de uma imagem se pode estabelecer suas propriedades de contraste e brilho. Segundo Gonzales & Woods (2008), se o histograma mostra que as variações de preto e branco se concentram em sua maioria em uma determinada faixa de valores e com frequências significativas, gerando assim um histograma estreito, como ilustra a Figura 6, então pode dizer que a imagem possui baixo contraste. Em contrapartida, se o histograma apresentar variações mais bem distribuídas e as freqüências em geral estabelecerem uma uniformidade, como ilustra a Figura 7, então é possível afirmar que se trata de uma imagem de alto contraste, ou seja, com uma grande variedade de tons de cinza. Em relação ao brilho dos *pixels* da imagem, quanto mais à direita do histograma as barras verticais se concentrarem maior será o brilho (mais clara) e quanto mais à esquerda menor será o brilho (mais escura), como ilustram as Figuras 8 e 9.

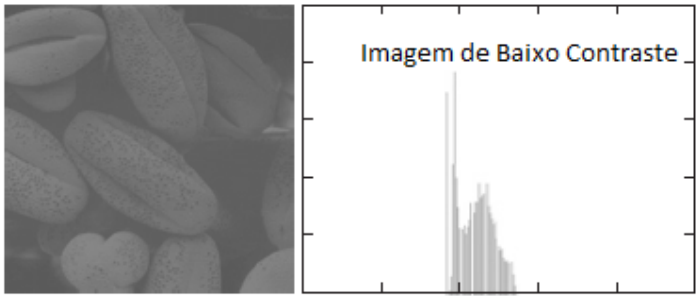


Figura 6: Imagem de baixo contraste.

Fonte: Traduzido de GONZALEZ, (2008)

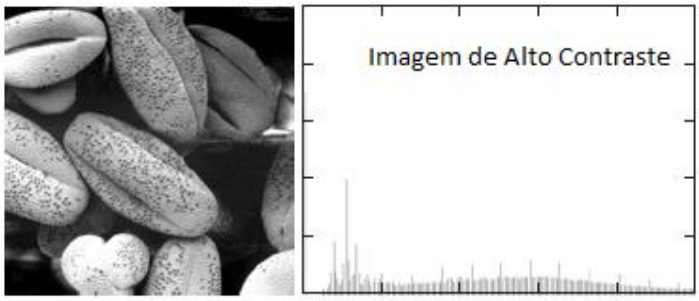


Figura 7: Imagem de alto contraste.

Fonte: Traduzido de GONZALEZ, (2008)

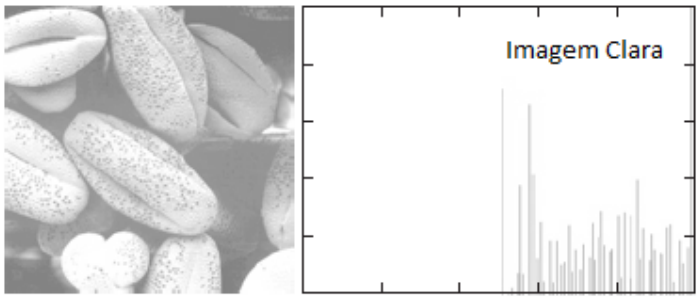


Figura 8: Imagem clara.

Fonte: Traduzido de GONZALEZ, (2008)

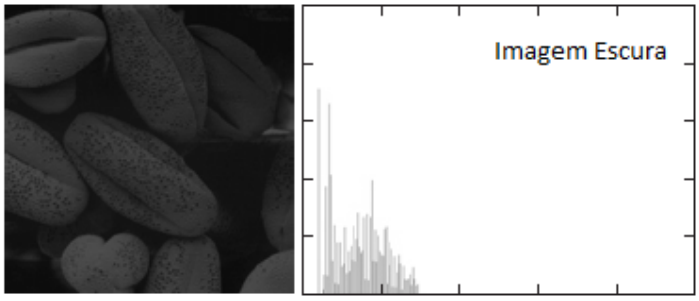


Figura 9: Imagem escura.

Fonte: Traduzido de GONZALEZ, (2008)

II. **Redução de Ruído:** geralmente um ruído consiste em um conjunto de *pixels* distribuídos na imagem que passam a impressão de granulação ou falta de qualidade da imagem. Consequentemente dificulta-se a extração de características da mesma. Alguns exemplos de ruído são: ruído sal e pimenta e ruído Gaussiano, como ilustra a Figura 10. Para reduzir o ruído, neste trabalho foi utilizado um filtro de freqüência passa-baixa no modo ideal, o qual será descrito em detalhes a seguir.

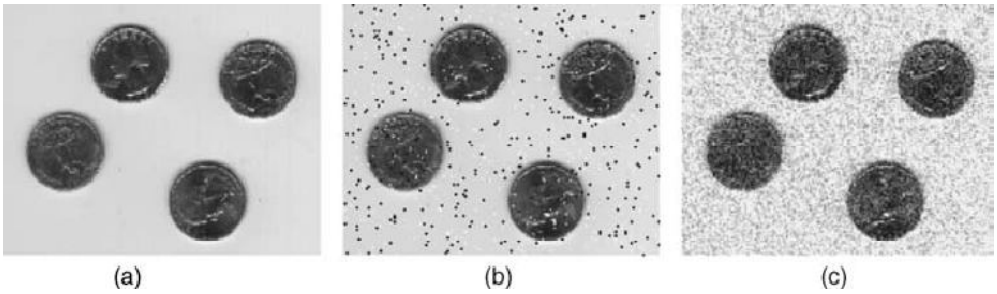


Figura 10: (a) Imagem original, (b) Imagem com ruído sal e pimenta, (c) Imagem com ruído Gaussiano

Fonte: SOLOMON, (2011)

2.3.6 Filtros de frequência

Conforme National Instruments (2005), filtros de frequência alteram valores de *pixels* de acordo com a periodicidade e distribuição espacial das variações de intensidade luminosa na imagem. De modo diferente aos filtros espaciais, os filtros de frequência não se aplicam diretamente à uma imagem espacial, mas à uma representação de frequência. A representação de frequência de uma imagem é obtida através da função Transformada Rápida de Fourier (FFT), a qual revela informações sobre a periodicidade e dispersão dos padrões encontrados na imagem de entrada. Com a imagem no domínio da frequência, cada componente (u,v) de frequência é individualmente ponderada, seguido da aplicação da FFT inversa, como ilustra a Figura 11 abaixo:

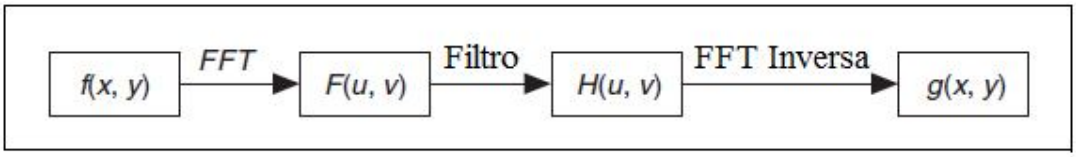


Figura 11: Sequência de operações utilizada em uma imagem com filtro de frequência aplicado

Fonte: Traduzido de NATIONAL INSTRUMENTS, (2005)

Em uma imagem, detalhes e arestas são associadas com médias e altas frequências espaciais, por estas representarem variações de níveis de cinza em pequenas distâncias. Variações mais longas ou regiões suaves são associadas com baixas frequências espaciais. Para filtrar frequências espaciais, é possível remover, atenuar ou destacar os componentes espaciais com que as mesmas se relacionam, conforme a seguir, segundo National Instruments (2005):

- **Filtro de frequência passa-baixa:** utilizado para atenuar altas frequências presentes na imagem. Este filtro suprime informações relacionadas com rápidas variações de intensidades luminosas na imagem espacial. Este filtro produz uma imagem onde ruídos, detalhes, textura e arestas são suavizadas.
- **Filtro de frequência passa-alta:** utilizado para atenuar baixas frequências. Este filtro suprime informações relacionadas com baixas variações de intensidades luminosas na imagem espacial. Neste caso, produz uma imagem onde bordas e detalhes da mesma são preservados, enquanto componentes suaves são removidos.

Neste trabalho, utilizou-se o filtro de frequência passa-baixa no modo truncado para realizar a redução de ruídos na imagem por atingir os resultados esperados satisfatoriamente. Segundo National Instruments (2005), o passa-baixa ideal remove uma frequência f se esta for maior do que a frequência de corte f_C. Isto é feito, multiplicando cada frequência f por um coeficiente C igual a 0 ou 1, dependendo de quando a frequência f é maior do que a frequência de corte f_C, como mostra a Figura 12 abaixo:

Se $f > f_c$

então $C(f) = 0$

senão $C(f) = 1$

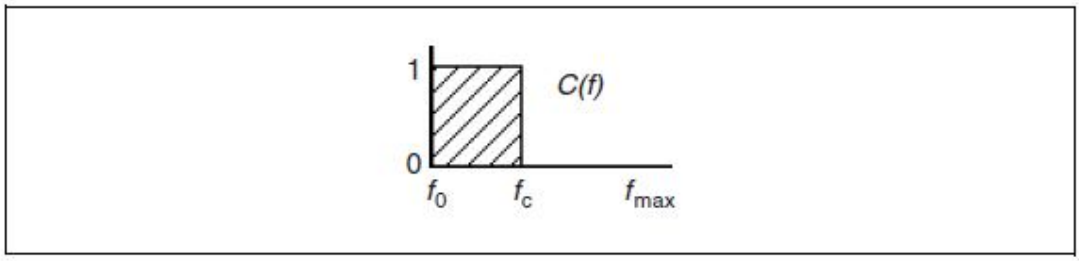


Figura 12: Sequência de operações utilizada em uma imagem com filtro de frequência passa-baixa ideal aplicado

Fonte: NATIONAL INSTRUMENTS, (2005)

A imagem da Figura 13 ilustra uma imagem FFT original e a Figura 16 ilustra a imagem FFT original com a aplicação do filtro passa-baixa ideal:

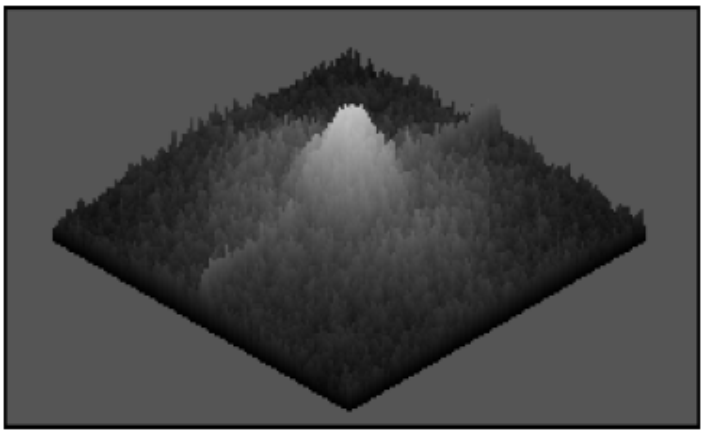


Figura 13: Imagem FFT original

Fonte: NATIONAL INSTRUMENTS, (2005)

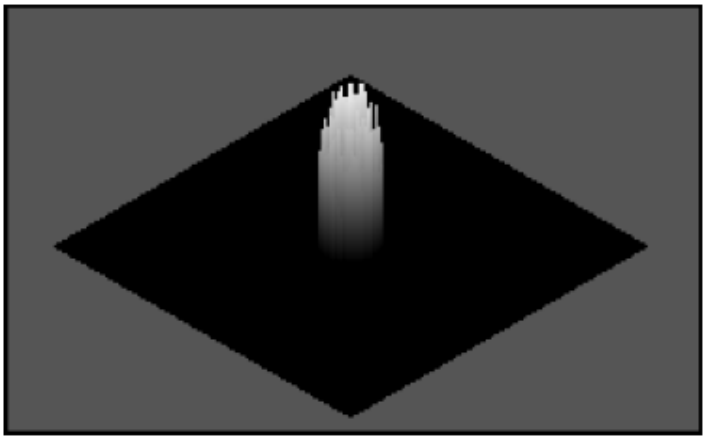


Figura 14: Imagem FFT original com filtro passa-baixo ideal aplicado

Fonte: NATIONAL INSTRUMENTS, (2005)

Após a aplicação do filtro passa-baixa ideal, as frequências espaciais fora da faixa de corte $[f_0, f_c]$ foram removidas. Ruídos, arestas, detalhes e texturas foram suprimidas ao máximo. Com isso, na Figura 14, apenas restou o pico central que se manteve idêntico ao pico da imagem original da Figura 13.

2.3.7 Segmentação

Segundo Marques Filho & Vieira Neto (1999), a segmentação consiste em dividir a imagem em suas diversas partes ou segmentos, ou seja, nos objetos de interesse que a compõem. O processo de separação acontece até que os objetos da cena analisada estejam isolados. Os algoritmos de segmentação são baseados em

uma das duas propriedades básicas do nível de intensidade luminosa das imagens: a descontinuidade e a similaridade. Na descontinuidade, pretende-se dividir a imagem baseando-se em mudanças abruptas do nível de intensidade luminosa, por exemplo, bordas de objetos na imagem. Já a similaridade, a idéia é dividir a imagem em regiões que são similares entre si de acordo com um critério pré-definido, como cor, textura ou luminosidade. As técnicas de segmentação utilizadas neste trabalho foram: detecção de bordas e limiarização (*thresholding*). Cada uma dessas técnicas estão introduzidas a seguir, conforme Gonzales & Woods (2008).

2.3.8 Detecção de bordas

A segmentação por detecção de bordas analisa regiões na imagem com mudanças abruptas do nível de intensidade luminosa. Existem três tipos comuns de descontinuidades: pontos, linhas e bordas. Para detectar cada uma dessas descontinuidades, são usadas configurações padrão de máscaras de convolução, ilustradas na Figura 15. Segundo Marques Filho & Vieira Neto (1999), a detecção de pontos isolados utiliza as máscaras de Laplace que destacam *pixels* brilhantes circundados por *pixels* mais escuros. Já a detecção de linhas utiliza máscaras especiais que encontram linhas horizontais, verticais e diagonais. Por fim, a detecção de bordas de objetos, que é a mais utilizada, aplica o operadores de Roberts, Prewitt e Sobel, baseados no conceito de gradiente. Segundo Pavim (2005), as bordas geralmente não se caracterizam por mudanças abruptas dos níveis de cinza (mudança radical no nível de intensidade luminosa), mas sim por mudanças suaves e contínuas (semelhantes ao sinal de uma rampa). Com isso, a intenção do filtro de gradiente é enfatizar esta mudança gradual e suave das bordas dos objetos.

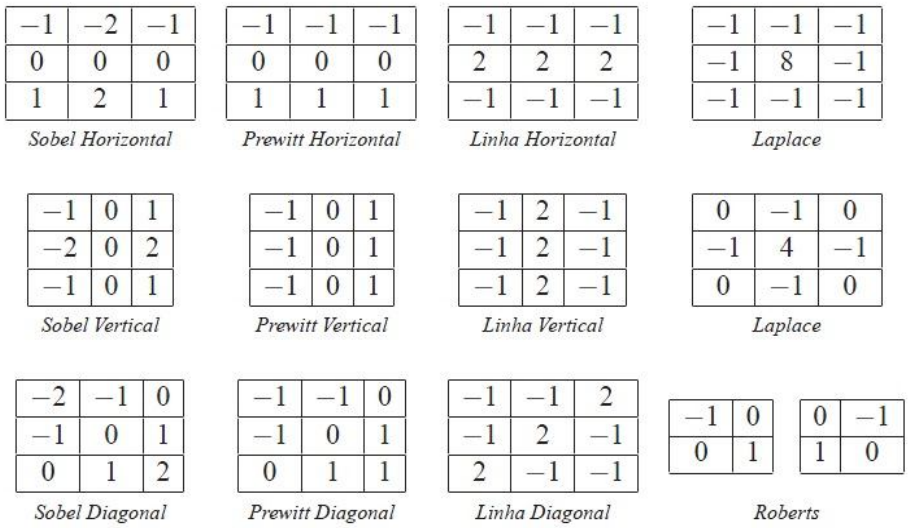


Figura 15: Configurações de máscaras para a detecção de descontinuidades.

Fonte: PAVIM, (2005)

2.3.9 Limiarização

Conforme Marques Filho & Vieira Neto (1999), esta técnica procura dividir a imagem quando esta apresenta duas classes, o fundo e o objeto de interesse. A operação de limiarização divide porções da imagem tendo um nível de cinza como limite. Cada valor de *pixel* da imagem é comparado com o limite. Se o valor estiver abaixo do limite, então o nível zero é atribuído ao *pixel*, já se o valor estiver acima do limite, então o nível um é atribuído ao *pixel*.

A limiarização pode ser dividida como simples ou múltipla. A simples é a mais comum e é aplicada geralmente em imagens que possuem duas porções que delimitam dois níveis de cinza, por exemplo, um objeto claro e um fundo escuro, ou o contrário. Como só há na imagem duas porções, então a escolha do limite é simples, considerando a utilização da técnica de iluminação adequada para tal caso. Já na múltipla, há múltiplos níveis de cinza, o que exige a utilização de técnicas mais avançadas que utilizam informações estatísticas e características de distribuição dos *pixels* da imagem para encontrar um valor limite, conforme Grassi (2005).

O processo de limiarização é definido como uma operação que envolve testes contra uma função T:

$$T = T[x, y, p(x, y), f(x, y)] \quad (1)$$

Onde *T* é o limite, isto é, um valor de limiar para o nível de intensidade luminosa ao qual se deseja realizar o ponto de corte da intensidade, $f(x,y)$ representa a intensidade luminosa de um ponto e $p(x,y)$ representa uma propriedade local deste ponto na imagem, como por exemplo, a média da intensidade de uma vizinhança centrada em (x,y), conforme Pavim (2005). A partir desta expressão, uma imagem limiarizada $g(x,y)$ é definida como:

$$g(x,y) = \begin{cases} 1 & \text{se } f(x,y) > T \\ 0 & \text{se } f(x,y) \leq T \end{cases} \quad (2)$$

Assim, se os objetos forem claros e o fundo for escuro, pontos da imagem rotulados com o valor 1 representam objetos, e pontos rotulados com o valor 0 representam o fundo da imagem. Quando *T* depende apenas de $f(x,y)$, a limiarização é chamada global. Se *T* depende também de $p(x,y)$, então a limiarização é chamada local. Caso *T* ainda dependa das coordenadas espaciais de x e y, é chamada de limiarização dinâmica ou adaptativa, conforme Gonzalez & Woods (2008).

Como exemplo de limiarização simples, em uma imagem de 256 tons de cinza, caso *T* = 30, todos os pontos da imagem entre 0 e 30 $(f(x,y) < T)$ se tornam informação do fundo da imagem (valor 0, ou cor preta). Já os demais valores a partir deste limiar $(f(x,y) > T)$ se tornam informação dos objetos da imagem (valor 1, ou cor branca). Assim, com a simples definição de um valor limiar ou de corte já é suficiente para dividir a imagem em duas regiões: fundo e objeto, conforme Pavim (2005).

Já para a limiarização múltipla, caso uma imagem $f(x,y)$ tenha dois valores distintos de limiar $(T_1 \text{ e } T_2)$, define-se que todos os valores em que $f(x,y) < T_1$ correspondem ao fundo da imagem, enquanto os valores em que $T_1 \leq f(x,y) < T_2$ correspondem a um determinado objeto, e quando $f(x,y) \geq T_2$ há outro objeto correspondente na cena, conforme Pavim (2005).

A Figura 16 ilustra estas duas situações acima:

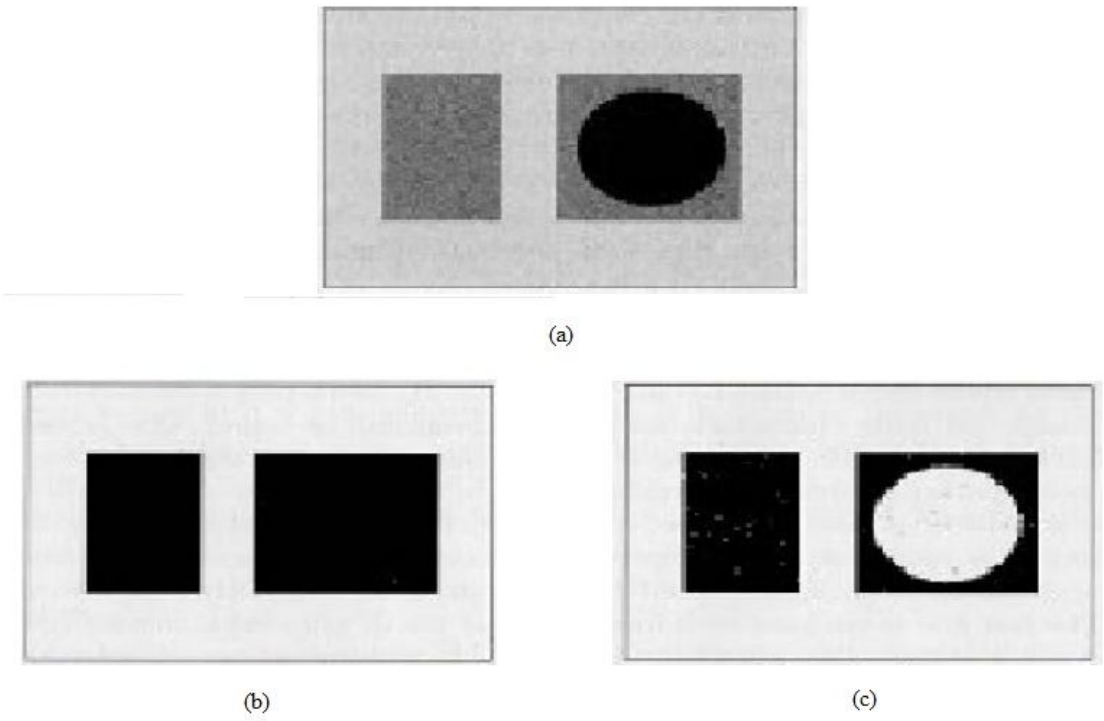


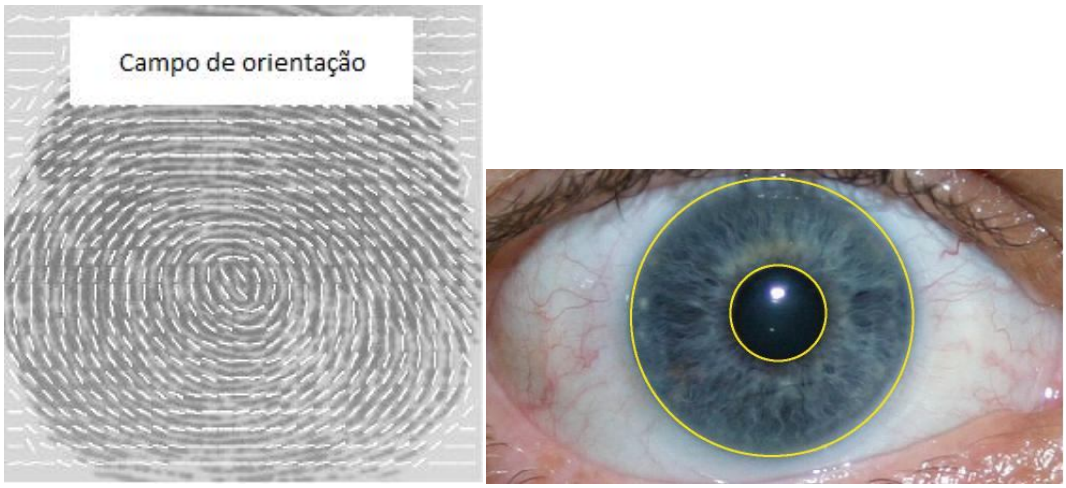
Figura 16: (a) Imagem original, (b) Imagem com limiarização simples, (c) Imagem com limiarização múltipla

Fonte: Adaptado de JANECZKO, (2010)

2.3.10 Representação e descrição

Após uma imagem ser segmentada em várias regiões, através de métodos de segmentação discutidos anteriormente, o resultado é um agregado de *pixels* que deve ser representado e descrito de maneira adequada, para que posteriormente seja processado em um ambiente computacional. Existem duas formas de representar uma região: considerar suas características externas (os limites ou fronteiras do objeto em questão), ou considerar suas características internas (os *pixels* que compõem a região). Uma representação externa é utilizada quando o foco é nas características geométricas, como em uma impressão digital ilustrada na Figura 17(a). Já a representação interna é utilizada quando o foco é nas propriedades da região, tais como cor e textura, como no processo de reconhecimento de íris ilustrado na Figura 17(b). A representação mais adequada para este trabalho é a externa, tendo em vista que o foco é localizar na imagem uma peça de geometria cilíndrica. A etapa de representação cumpre a tarefa de fornecer dados adequados ao computador. Já a descrição, utiliza descritores que baseados

na forma de representação escolhida, devem extrair características suficientes da região e seu respectivo objeto para distinguir classes de objetos, de preferência independente de variações de tamanho, rotação ou translação. Por exemplo, uma região pode ser representada por suas características externas e ser descrita considerando o comprimento ou o número de concavidades do objeto analisado, conforme Gonzalez & Woods (2008).



(a) (b)

Figura 17: (a) Impressão digital e suas características externas, (b) Reconhecimento de íris e suas características internas

Fonte: (a) Traduzido de JAIN, (1997), (b) MITSUBISHI ELECTRIC, (2007)

2.3.11 Reconhecimento e interpretação

Segundo Gonzales & Woods (2008), reconhecimento é o processo de atribuir um rótulo para um objeto baseado nas informações extraídas pelos descritores. A tarefa de interpretação, por outro lado, consiste em atribuir um significado a um conjunto de objetos reconhecidos, conforme Marques Filho & Vieira Neto (1999). O rótulo representa a classificação do objeto reconhecido. Conforme Solomon (2011), o objetivo da classificação em processamento de imagens é identificar aspectos característicos, padrões ou estruturas na imagem e usar isto para atribuí-las a uma classe específica. Um exemplo de tarefas de classificação é identificar células em uma lâmina histológica atribuindo rótulos como ‘normais’ ou ‘anormais’. Conceitos de sistemas de classificação são apresentados no apêndice A.

2.3.12 Base de conhecimento

Segundo Jardim (2006), para reconhecer um objeto em uma imagem é necessário conhecer previamente alguma característica desse objeto, como por

exemplo, sua geometria. Nesse contexto, surge a função da base de conhecimento, que é a de armazenar esta característica do objeto em forma de dados, os quais serão interpretados pelas etapas de processamento de imagens ilustradas na Figura 4. Com o objetivo de reconhecer esta característica, pode-se comparar uma imagem plana bidimensional com um modelo armazenado na base de conhecimento. Esta prática é chamada de **reconhecimento baseado em modelos**. T

As técnicas mais clássicas para o reconhecimento de objetos baseado em modelos são, conforme Jardim (2006):

- **Árvores de interpretação:** técnica que engloba a classe de algoritmos que se baseiam na identificação de características. Identificam partes de um objeto em uma imagem, isto é, todas as regiões da imagem que tenham forma de um modelo de características conhecido (que esteja armazenado na base de conhecimento).

- **Identificação baseado em aparências:** técnica que consiste nas imagens propriamente ditas dos objetos analisados e não de suas características. Identificam se em partes de uma imagem encontram-se partes de um determinado objeto, ou seja, identificar um objeto baseado em sua aparência. Para que isto seja possível, é necessário que um conjunto de imagens do objeto, em diversos ângulos, esteja armazenado na base de conhecimento.

- **Invariantes:** são algoritmos que identificam objetos, utilizando-se de características geométricas invariantes de modelos armazenados na base de conhecimento. O objetivo desta técnica é armazenar as características invariantes dos objetos, para sua posterior identificação em determinada imagem. Características invariantes são propriedades funcionais de configurações geométricas que não se alteram sob certas classes de transformações. Tais como, comprimento, área, circularidade, razão de comprimento e ângulo e a razão das razões de comprimentos (*cross-ratio*).

2.4 Conceitos de topografia e geodésia

Em função do sistema de visão robótico deste trabalho necessitar trabalhar com dois sistemas de coordenadas distintos posicionados de forma que seus eixos

X e Y não são paralelos e as origens estão em locais diferentes e desalinhadas (Figura 18), é necessário correção de coordenadas. Por escolha do autor, foram aplicados conceitos de topografia e geodésia para efetuar a rotação dos sistemas para que os eixos X e Y fiquem paralelos e calcular a translação dos sistemas para que a origem dos mesmos seja compatibilizada.

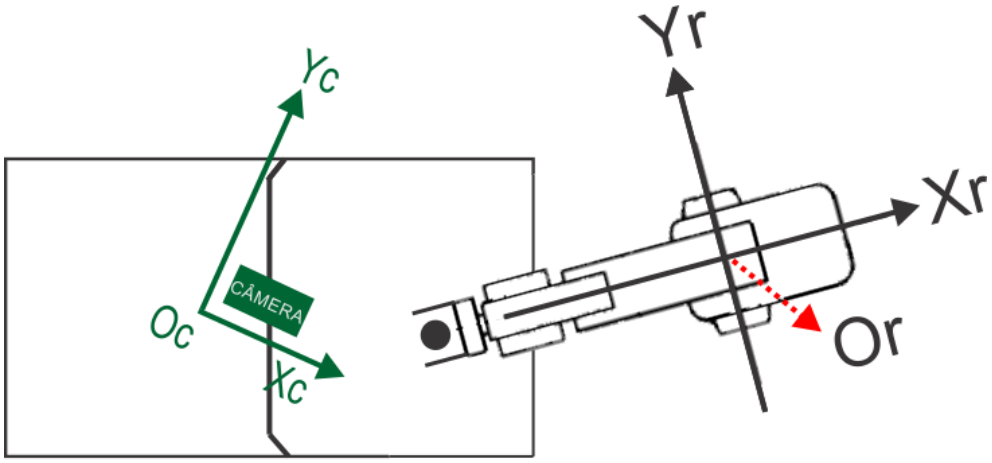


Figura 18: Sistema de visão robótico com sistemas de coordenadas distintos sem a aplicação de rotação e translação

Fonte: Autor

Topografia é a ciência que trata da determinação das dimensões e contornos (ou características tridimensionais) da superfície física da Terra, conforme McCormac (2007).

Levantamentos geodésicos são aqueles que consideram a curvatura da superfície física da Terra (a Terra é um elipsoide cujo raio no equador é cerca de 21,7 km maior que o raio polar), conforme McCormac (2007).

Tanto em topografia como em geodésia os lados de uma poligonal levantada (polígono formado pelos lados medidos de um terreno) precisam ter uma orientação definida em relação à linha norte/sul que pode ser verdadeira ou magnética. A verdadeira (norte verdadeiro) possui como referência, se georreferenciado através de um GPS, o meridiano (que são círculos máximos que passam pelos pólos da Terra) do elipsóide, o qual é adotado para a forma da Terra, que é obrigatório para trabalhos oficiais. O elipsoide é adotado para representar a forma da Terra, embora a forma real da Terra seja uma Figura chamada geoide. Já a magnética possui como referência a bússola.

2.4.1 Azimute

Segundo McCormac (2007), um termo comum utilizado para designar uma direção de uma linha é azimute. O azimute de uma direção é definido pelo ângulo em sentido horário do extremo norte ou sul do meridiano de referência para a linha em questão. Para levantamentos planos comuns (utilizados em áreas relativamente pequenas onde se considera a Terra como plana), os azimutes são geralmente medidos à partir o lado norte do meridiano. O valor de um azimute pode variar de 0 a 360°.

Em topografia e geodésia, os quadrantes do círculo trigonométrico da matemática são contados no sentido horário, da mesma forma os azimutes também são contados no sentido horário a partir da direção norte/sul que por sua vez coincide com o eixo das ordenadas ou Y de um sistema cartesiano. Desta forma, em levantamentos topográficos de terrenos (levantamentos planos comuns) são medidos ângulos e distâncias entre pontos que formam um alinhamento ou lados de uma poligonal os quais devem ter uma orientação segundo a linha norte/sul (azimute), conforme Borges & Cordeiro (2012).

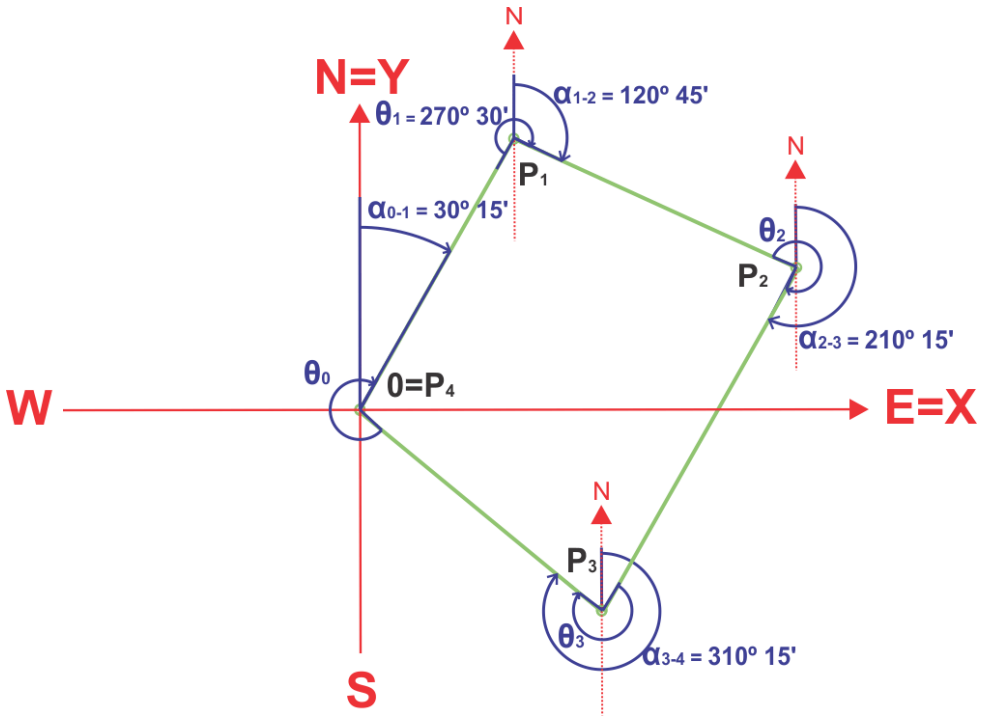


Figura 19: Azimutes de uma poligonal fechada

Fonte: Autor

Para calcular os azimutes dos alinhamentos ou lados de uma poligonal que forma as divisas de um terreno, a partir dos ângulos formados entre esses alinhamentos medidos no campo, como ilustra a Figura 19, é necessário medir também um azimute inicial (primeiro lado ou alinhamento da poligonal, na Figura 19, α_{0-1}) e os demais azimutes (α_{1-2}, α_{2-3}, α_{3-4}) são calculados da seguinte forma:

Exemplo: considerando os dados da Figura 19, onde o ângulo externo θ_1 formado pelos alinhamentos $0 - P_1$ e $P_1 - P_2$ (onde 0 é a origem do sistema) é igual a 270º30', cujos alinhamentos podem ser os lados de uma poligonal com caminhamento (trajetória $0 - P_4$), o qual inicia no ponto 0 e passa pelos pontos P_1, P_2, P_3 e P_4, onde P_4 é igual a 0. Considerando ainda que o azimute inicial α_{0-1} é determinado em relação à linha norte/sul.

O azimute do alinhamento $0 - P_1$ chamado α_{0-1}, indicado na Figura 20, pode ser obtido com a bússola, ou através de georreferenciamento que determina as coordenadas (x,y) do ponto inicial 0 (X_0,Y_0) e do ponto seguinte P_1 (X_1,Y_1), com GPS de precisão (geodésico). Com as coordenadas obtidas é calculado o azimute do alinhamento da seguinte forma:

$$\alpha_{0-1} = arc\tan(\varphi) \tag{3}$$

Fonte: BORGES & CORDEIRO, (2012)

$$\tan\varphi = (X_1 - X_0) / (Y_1 - Y_0) \tag{4}$$

Fonte: BORGES & CORDEIRO, (2012)

Onde φ determina o azimute pelo conjunto de sinais, conforme tabela 2 abaixo:

$(X_1 - X_0) > 0$ e $(Y_1 - Y_0) > 0$	$\alpha_{0-1} = \varphi$
$(X_1 - X_0) > 0$ e $(Y_1 - Y_0) < 0$	$\alpha_{0-1} = 180 - \varphi$
$(X_1 - X_0) < 0$ e $(Y_1 - Y_0) < 0$	$\alpha_{0-1} = 180 + \varphi$
$(X_1 - X_0) < 0$ e $(Y_1 - Y_0) > 0$	$\alpha_{0-1} = 360 - \varphi$

Tabela 2: Combinação de sinais para determinação do azimute
Fonte: BORGES & CORDEIRO, (2012)

Nota-se que este cálculo é utilizado somente para calcular o azimute à partir das coordenadas dos dois pontos que formam o alinhamento.

A partir do azimute inicial α_{0-1} são calculados os azimutes dos alinhamentos subseqüentes utilizando o ângulo formado pelo alinhamento anterior e o alinhamento seguinte, seguindo os passos abaixo, conforme Borges & Cordeiro (2012):

1º passo: Somar α_{0-1} com θ_1 = 30º 15' + 270º 30' = 300º 45'

2º passo: Verificar se a soma do passo 1 passou de 360º, se passar, subtrai 360º;

3º passo: Verifica se o resultado da soma do passo 1 é maior ou menor do que 180º: no caso é maior, portanto subtraio 180º, com isso o azimute do alinhamento α_{1-2} = 120º 45'. Se fosse menor, teria que somar 180º.

Importante: observar que o primeiro e o terceiro passo sempre existem, o segundo só existe se a primeira soma passar de 360º.

Tais passos são resumidos na equação 5 a seguir:

$$\alpha_{1-2} = \begin{cases} se\ \alpha_{0-1} + \theta_1 > 360 = (\alpha_{0-1} + \theta_1) - 360 \\ se\ \alpha_{0-1} + \theta_1 > 180 = (\alpha_{0-1} + \theta_1) - 180 \\ se\ \alpha_{0-1} + \theta_1 < 180 = (\alpha_{0-1} + \theta_1) + 180 \end{cases} \quad (5)$$

Fonte: BORGES & CORDEIRO, (2012)

Os demais azimutes na sequência são calculados sempre seguindo esses três passos que valem para qualquer situação. Os ângulos à direita (sentido horário) formados entre os alinhamentos podem ser internos ou externos, dependendo do sentido do caminhamento.

Lembrando que este cálculo é utilizado apenas para calcular o azimute à partir de um azimute inicial conhecido e um ângulo formado entre o alinhamento do azimute inicial e o alinhamento subsequente.

2.4.2 Transporte de coordenadas e rotação e translação de sistemas

Em topografia é comum encontrar-se levantamentos topográficos feitos com orientação magnética, isto é, não georreferenciados, surgindo daí a necessidade de transportar coordenadas de pontos georreferenciados, chamados pontos geodésicos, para transformar ou compatibilizar os sistemas. Quando não se dispõe de equipamentos que permitam fazer com relativa facilidade o georreferenciamento no local, como o GPS geodésico, por se tratar de equipamentos de alto custo, conforme Borges & Cordeiro (2012).

Para o presente trabalho, em uma analogia comparativa com os sistemas de coordenadas do robô e da câmara, o sistema de coordenadas do robô, pode-se considerar o sistema georreferenciado e o sistema da câmara pode-se considerar o sistema não georreferenciado ou com orientação magnética, isto é, com a bússola, o que leva a variações na orientação.

Como os sistemas coordenados do robô e da câmera não estão compatibilizados (eixos coordenados não paralelos e origens em locais diferentes e desalinhadas), é necessário realizar a correção dos sistemas. Em topografia e geodésia as operações podem ser resumidas pelas Figuras 20 e 21, cujos cálculos são apresentados no apêndice B.

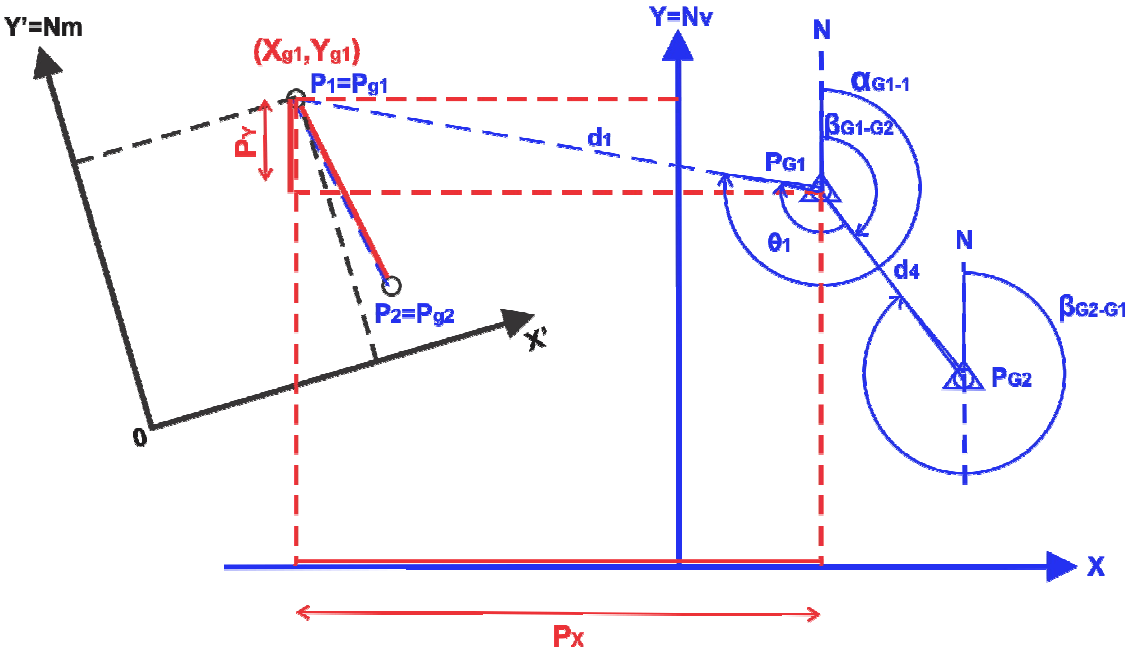


Figura 20: Transporte das coordenadas do ponto P_1
Fonte: Autor

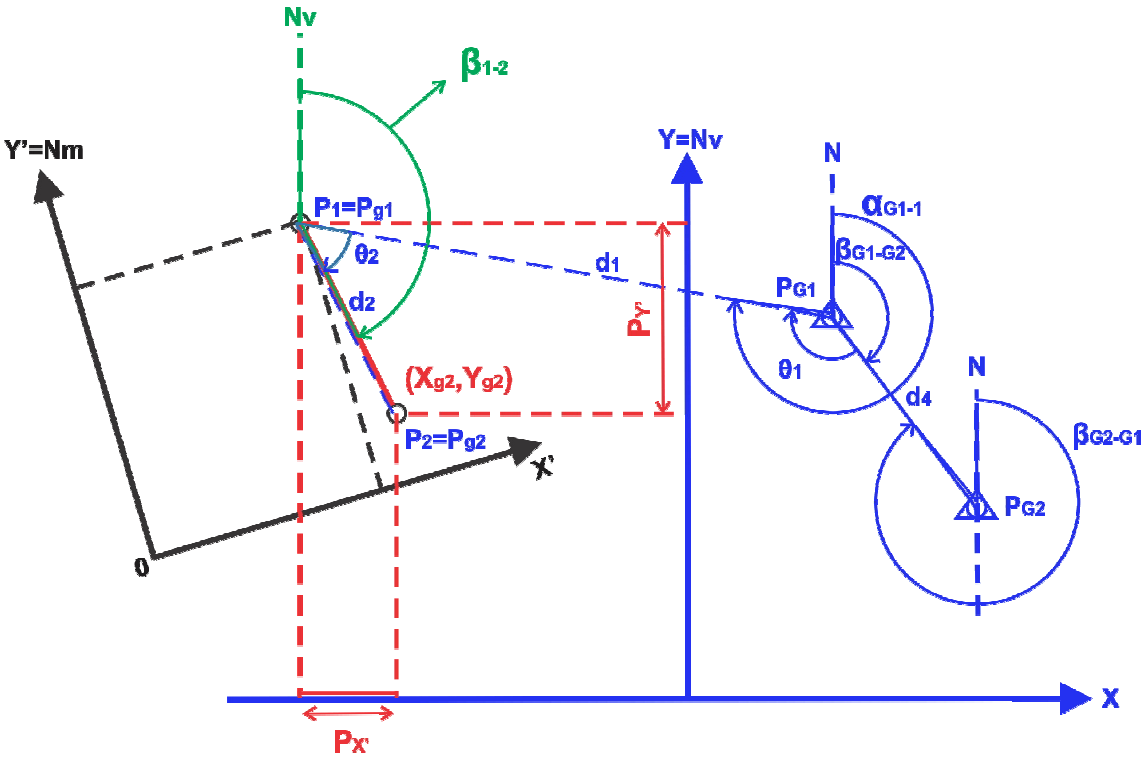


Figura 21: Transporte das coordenadas do ponto P_2
Fonte: Autor

No caso deste trabalho, as coordenadas do sistema do robô ("georreferenciado") já são conhecidas por serem pré-estabelecidas na etapa de calibração. E as coordenadas do sistema da câmera (não referenciado) são obtidas através do programa de processamento de imagens desenvolvido no *software* Vision Builder. Como consequência não é necessário realizar o transporte de coordenadas para os dois pontos pré-estabelecidos, porém é necessário realizar o transporte de coordenadas para esses pontos e a origem do sistema da câmera com relação ao sistema do robô. Isto é, as coordenadas da origem do sistema da câmera são encontradas através do transporte de coordenadas, como explicado anteriormente. Tais coordenadas são necessárias para realizar a translação entre os dois sistemas, o que os tornam compatíveis entre si.

2.4.3 Rotação

A rotação θ, ilustrada na Figura 22, entre dois sistemas de coordenadas diferentes pode ser calculada pela diferença dos azimutes calculados no sistema georreferenciado (orientado pela linha norte/sul verdadeira - Nv) e não

georreferenciado (orientado pela linha norte/sul magnética - Nm) ou na comparação com o trabalho (estabelecido pelo autor), no sistema do robô e da câmara. O cálculo de α_{1-2} é realizado da mesma forma apresentada na seção 2.5.3.

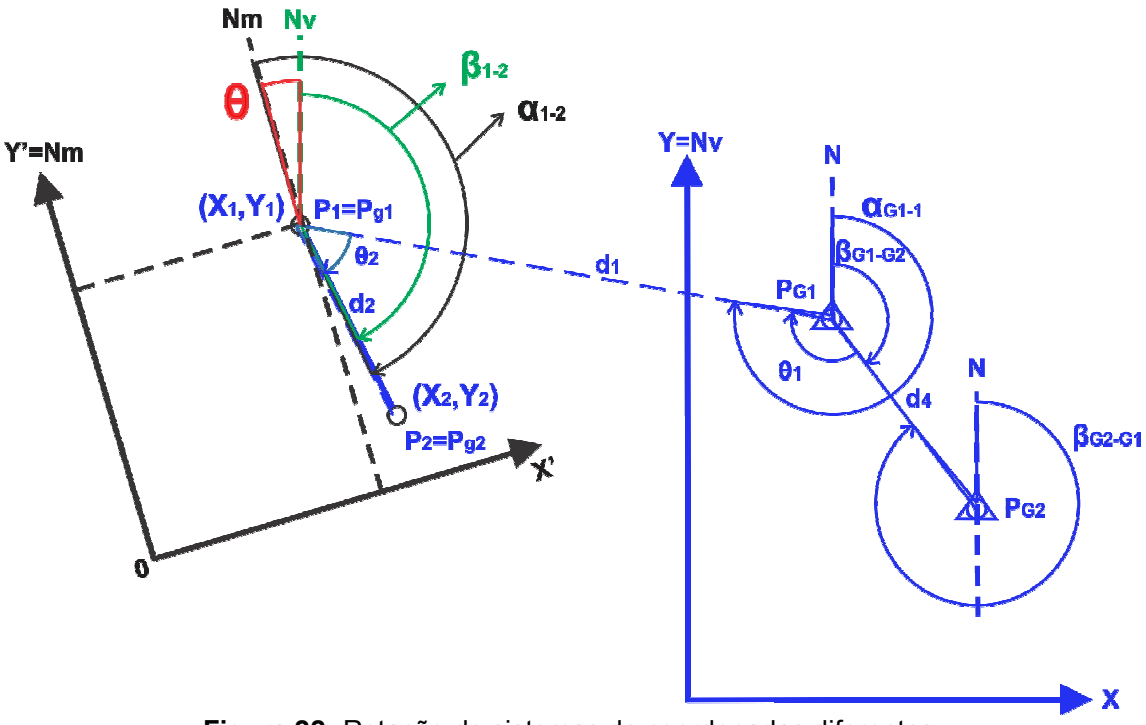


Figura 22: Rotação de sistemas de coordenadas diferentes
Fonte: Autor

Com isto, o ângulo de rotação θ é calculado pela diferença do maior azimute α_{1-2} (destacado em preto) pelo menor azimute β_{1-2} (destacado em verde), conforme abaixo:

$$\theta = \alpha_{1-2} - \beta_{1-2} \tag{6}$$

Fonte: BORGES & CORDEIRO, (2012)

Para este trabalho, utiliza-se o ângulo de rotação θ para que o sistema da câmera (não referenciado) seja paralelo ao sistema do robô (“georreferenciado”).

2.4.4 Cálculo de erro

A maior preocupação em topografia é a precisão do trabalho, conforme McCormac (2007). Segundo o mesmo autor, existem três fontes de erros: do operador, de instrumentos e da natureza. Consequentemente, os erros nas medições são geralmente ditos como operacionais, instrumentais e naturais. Alguns

erros, no entanto, não se ajustam claramente a uma dessas categorias e podem ocorrer devido a uma combinação de fatores.

Erros operacionais: ocorrem porque nenhum topógrafo tem sentido perfeito de visão e tato. Por exemplo: ao estimar a parte fracionária de uma escala, o topógrafo não consegue ler perfeitamente, e sempre a estimativa será um pouco maior ou menor;

Erros instrumentais: ocorrem porque os instrumentos não podem ser fabricados de forma perfeita, além do desgaste pelo uso ao longo do tempo;

Erros naturais: são causados por temperatura, vento, umidade, variações magnéticas, dentre outras causas.

Em topografia, para ter o controle dos erros cometidos na medição é necessário medir a posição de um ponto de diferentes posições ou o que mais usual, formar uma poligonal fechada. Isto é, percorre-se o perímetro de um terreno medindo-se os ângulos entre os alinhamentos ou lados da poligonal formada, bem como as distâncias ou comprimento desses lados, partindo-se de um ponto inicial e retornando ao mesmo ponto. Os erros cometidos nas medidas, desde que estejam dentro das tolerâncias, conforme normas, são corrigidos. Tais cálculos são demonstrados no apêndice F.

Para compatibilizar os dois sistemas, foi necessário calcular além da rotação já citada, também as projeções rotacionadas (por exemplo, na Figura 24 do próximo tópico, $P_{Xr'}$ e $P_{Yr'}$ destacadas em amarelo) para ser possível realizar a translação dos dois sistemas (onde se obtém as coordenadas da origem do sistema da câmera no sistema do robô). Tal operação será descrita a seguir.

2.4.5 Translação

A translação dos sistemas é feita recalculando-se as coordenadas do sistema não georreferenciado ou da câmera, no sistema georreferenciado ou do robô. Para tanto, após ser feita a rotação da forma já mostrada, onde se encontrou o ângulo θ, é necessário calcular também as coordenadas da origem (X_0, Y_0) do sistema não georreferenciado ou da câmara no sistema do robô (“georreferenciado”), compatibilizando desta forma os dois sistemas. Para que tais coordenadas sejam encontradas, deve-se seguir os cálculos a seguir.

Primeiramente, deve-se calcular o azimute (realizado da mesma forma apresentada na seção 2.5.3) e a distância do alinhamento $P_0 - P_1$ (α_{0-1}, destacado em preto na Figura 23) da seguinte forma:

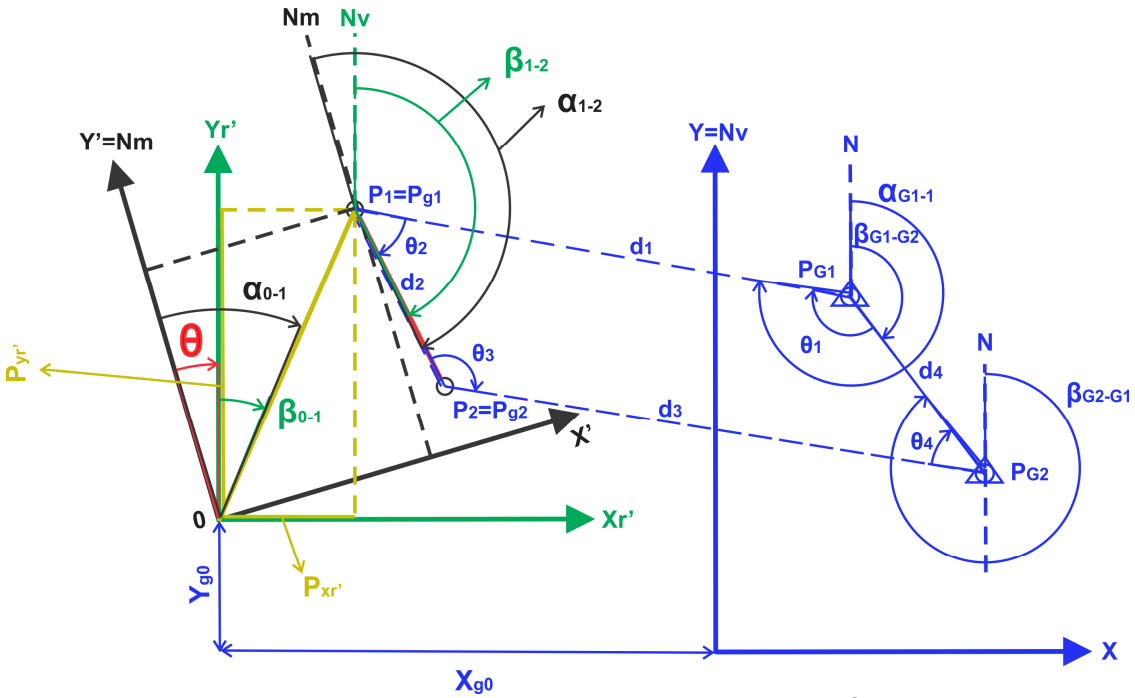


Figura 23: Translação de coordenadas de sistemas diferentes
Fonte: Autor

- **DISTÂNCIA ENTRE DOIS PONTOS**

A distância entre dois pontos pode ser calculada pelas suas coordenadas utilizando a fórmula de Pitágoras a seguir:

$$d = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \quad (7)$$

Fonte: BORGES & CORDEIRO, (2012)

Onde:

d = distância entre dois pontos de coordenadas X e Y

X_1 e Y_1 = coordenadas da origem do alinhamento formado pelos dois pontos

X_2 e Y_2 = coordenadas do destino do alinhamento

Com isso, a distância do alinhamento $P_0 - P_1$ é determinada conforme a seguir:

$$D_{O-1} = \sqrt{(X_1 - X_o)^2 + (Y_1 - Y_o)^2} \quad (8)$$

Fonte: BORGES & CORDEIRO, (2012)

Com isso, pode-se calcular o β_{0-1} da seguinte forma (onde θ é a rotação calculada anteriormente):

$$\beta_{0-1} = \alpha_{0-1} - \theta \quad (9)$$

Fonte: BORGES & CORDEIRO, (2012)

Observa-se na Figura 24, um sistema de coordenadas (*Xr', Yr'*) destacado na cor verde que é o resultado da rotação do sistema não georreferenciado para o georreferenciado. As projeções rotacionadas em relação ao alinhamento $P_0 - P_{g1}$ (destacadas em amarelo) $P_{Xr'}$ e $P_{Yr'}$ são calculadas como segue:

$$P_{Xr'} = D_{O-1}\ x\ sen(\beta_{0-1}) \quad (10)$$

Fonte: BORGES & CORDEIRO, (2012)

$$P_{Yr'} = D_{O-1}\ x\ cos(\beta_{0-1}) \quad (11)$$

Fonte: BORGES & CORDEIRO, (2012)

Com essas projeções é possível calcular as coordenadas da origem *P0* no sistema georreferenciado, que passou a ser chamado de P_{g0}, conforme a seguir:

$$X_{g0} = P_{Xr'} + X_{g1} \quad (12)$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_{g0} = P_{Yr'} + Y_{g1} \quad (13)$$

Fonte: BORGES & CORDEIRO, (2012)

Demais coordenadas do sistema não georreferenciado são calculadas somando-se algebricamente as coordenadas do ponto P_{g0} (X_{g0}, Y_{g0}) com as coordenadas do sistema não georreferenciado.

No caso da câmera e do robô, uma vez conhecida as coordenadas da origem do sistema da câmera no sistema do robô, isto é o ponto P_{g0} (X_{g0}, Y_{g0}), as demais coordenadas desconhecidas do sistema da câmera são obtidas somando-se

algebricamente as coordenadas rotacionadas com as coordenadas da origem (X_{g0},Y_{g0}). Tal processo é explicado em detalhes no capítulo de desenvolvimento.

3. DESENVOLVIMENTO DO SISTEMA

Neste capítulo são descritos todos os procedimentos e etapas realizadas durante a fase de desenvolvimento e conclusão deste projeto.

Foi desenvolvido um sistema de visão robótico, como ilustra a Figura 24. O computador processa todas as informações geradas pelo programa de processamento de imagens e realiza os cálculos necessários para que seja possível gerar instruções corretas para o robô manipular a peça. Como resultado, o posicionamento da peça a ser manipulada é reconhecida e identificada. Assim, instruções exatas e precisas são geradas para que o braço robótico atuante no processo manipule a peça em qualquer ponto em sua área de trabalho e dentro do campo de visão da câmera. Posteriormente, o braço robótico descarrega a peça em um ponto fixo pré-estabelecido e localizado dentro de seu espaço de trabalho.

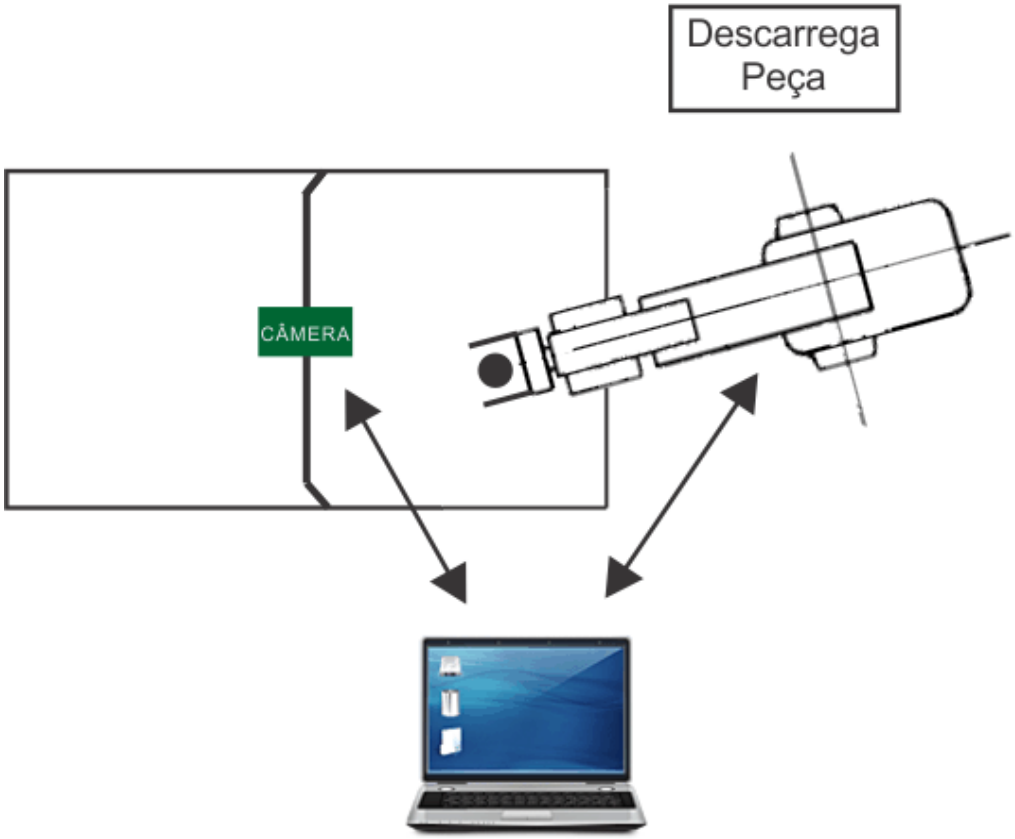


Figura 24: Sistema de visão robótico

Fonte: Autor

O sistema de visão robótico desenvolvido é dividido pelos itens a seguir:

- Definições do sistema
- Equipamentos e programas utilizados
- Iluminação
- Programa de processamento de imagens
- Programa de manipulação

3.1 Definições do sistema

Após serem realizados testes iniciais com a câmera e o programa de processamento de imagens, foram estabelecidas as seguintes definições:

- **Dimensões utilizadas:** foi estipulado que o sistema trabalharia apenas em duas dimensões X e Y. Sendo assim, a dimensão Z é fixa no programa de manipulação, o que exige que a peça a ser manipulada possua sua altura pré-definida.

  Justificativa: facilitar os cálculos realizados pelo programa de manipulação.
- **Câmera:** foi definida a utilização de apenas uma câmera para a monitoração de todo o plano em que se localiza o objeto a ser manipulado. A câmera é posicionada diretamente acima do plano do objeto, como mostra a Figura 25**.**

Figura 25: Câmera posicionada acima do plano do objeto

Fonte: Autor

Justificativa: como são utilizadas apenas as dimensões X e Y, apenas uma câmera é suficiente para determinar a posição da peça.

- **Objeto de manipulação:** uma peça de formato cilíndrico com dimensões e geometria conhecida.

  Justificativa: como a dimensão Z é fixa, a altura da peça deve ser previamente conhecida para o correto funcionamento do sistema. E para facilitar o processamento de imagens, optou-se pelo formato cilíndrico, o qual possui o processo de reconhecimento menos complexo do que um formato não conhecido. Também, a garra do robô é feita para manipular cilindros.

- **Posicionamento da câmera:** é obrigatório que a câmera esteja paralela ao plano que se encontra a peça a ser manipulada, isto é, paralela à mesa.

  Justificativa: quando isto não acontece, as coordenadas da peça lidas pelo programa de processamento de imagens são irreais. Isto ocorre, porque quando o campo de visão da câmera não está paralelo com o plano que se encontra a peça, a câmera gerará coordenadas que não coincidem com a posição real da peça sobre a mesa. Já que as coordenadas se referem a um plano diferente (campo de visão da câmera) do plano real da peça (mesa), o que não acontece se esses planos forem coincidentes.

- **Número de peças:** o sistema de visão robótica reconhece, localiza e manipula apenas uma peça por vez.

  Justificativa: caso mais peças fossem analisadas ao mesmo tempo, a complexidade do sistema aumentaria demasiadamente, exigindo a implementação de algoritmos computacionais mais complexos, o que possivelmente atrasaria o desenvolvimento deste trabalho.

- **Flexibilidade:** o sistema de coordenadas do robô não precisa estar paralelo ao sistema de coordenadas da câmera, isto é, a origem e os eixos coordenados do robô não precisam estar alinhados com a origem e os eixos coordenados da câmera, como ilustra a Figura 26.

  Justificativa: dessa forma, quem opera o sistema não precisa garantir que os sistemas coordenados do robô e da câmera estejam exatamente alinhados e paralelos, como ilustra a Figura 27 (considerando que o eixo Z de ambos os sistemas estejam alinhados). Assim, o sistema de coordenadas da câmera pode estar orientado para qualquer direção e fixo em qualquer lugar, desde que o seu

campo de visão esteja dentro da área de trabalho do robô. Isto gera grande flexibilidade ao sistema, já que o mesmo funciona em variadas situações.

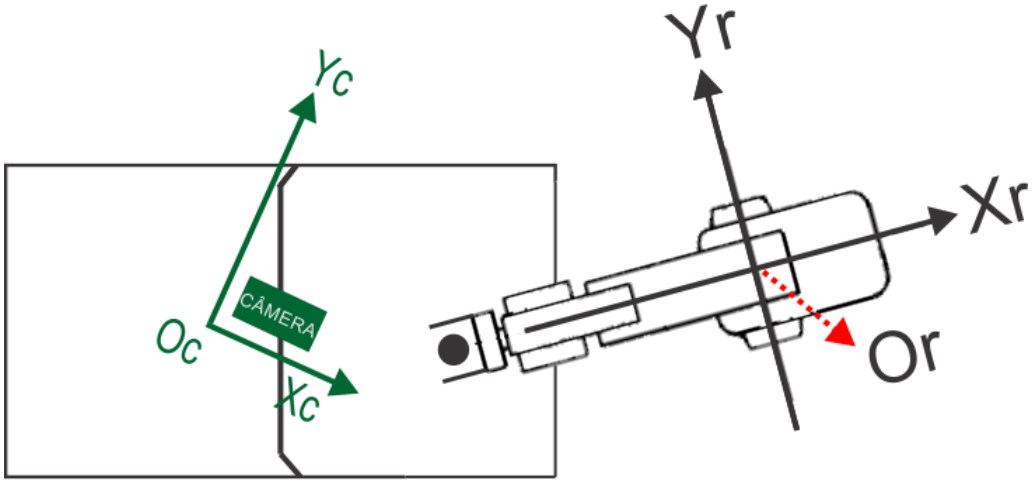


Figura 26: Sistemas coordenados não paralelos e desalinhados

Fonte: Autor

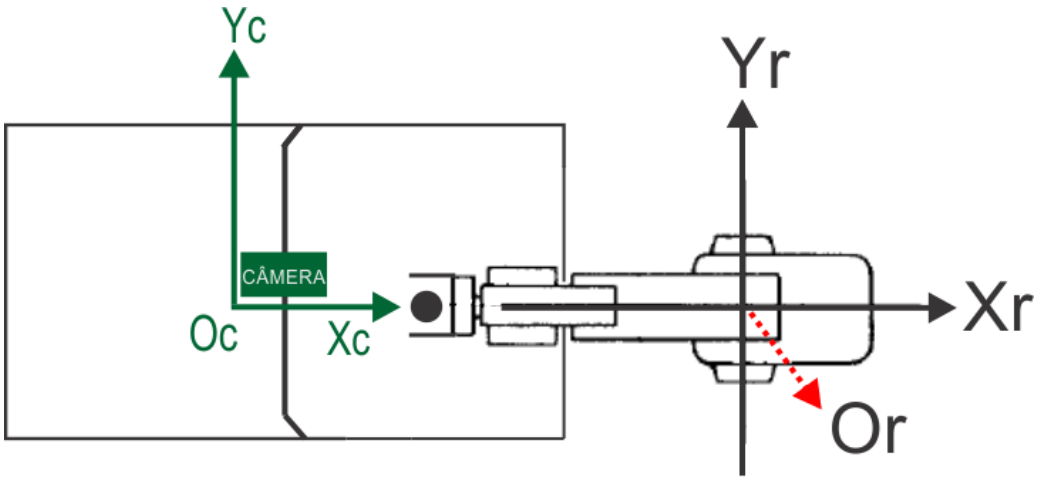


Figura 27: Sistemas coordenados paralelos e alinhados

Fonte: Autor

3.2 Equipamentos e programas utilizados

Foram selecionados equipamentos em função da disponibilidade de recursos e infraestrutura instalada no Laboratório de Automação e Sistemas Inteligentes da Manufatura da Universidade Tecnológica Federal do Paraná, campus Curitiba.

3.2.1 Câmera

Foi utilizada a *webcam Logitech HD Webcam C270*. Decidiu-se utilizar uma *webcam* ao considerar a vantagem de dispensar uma placa de captura de imagem por possuir uma conexão USB (*Universal Serial Bus*). Em contrapartida foi necessário adquirir uma *webcam* de alta definição, para que sejam geradas imagens de alta qualidade e facilitar a posterior análise das mesmas na etapa de processamento de imagens.

3.2.2 Mesa e suporte da câmera

A mesa tem como função, servir de apoio para a peça a ser analisada pelo programa de processamento de imagens e posteriormente ser manipulada através de instruções geradas pelo programa de manipulação, e de suporte para a câmera localizada diretamente acima do plano da mesma. Sendo assim, inicialmente foram feitos testes experimentais do programa de processamento de imagens em mesas comuns existentes no laboratório de FMS da UTFPR, campus Curitiba. Os testes foram satisfatórios, porém a dependência de ter uma mesa disponível no laboratório e com uma altura que satisfaça às necessidades do sistema, dificultavam o desenvolvimento do projeto. Com isso, foi construída uma mesa de madeira maciça com altura definida de 75 cm, largura 40 cm e comprimento 1,20 m. Os pés da mesa não são acoplados à mesma. São utilizados cavaletes, o que facilita o transporte da mesa. Para facilitar a etapa de processamento das imagens adquiridas da câmera, a superfície da mesa foi coberta por uma cartolina, cor preta. Assim, o fundo das imagens geradas é identificado com maior facilidade pelo programa de processamento e consequentemente a peça a ser manipulada também. A mesa é ilustrada na Figura 28.

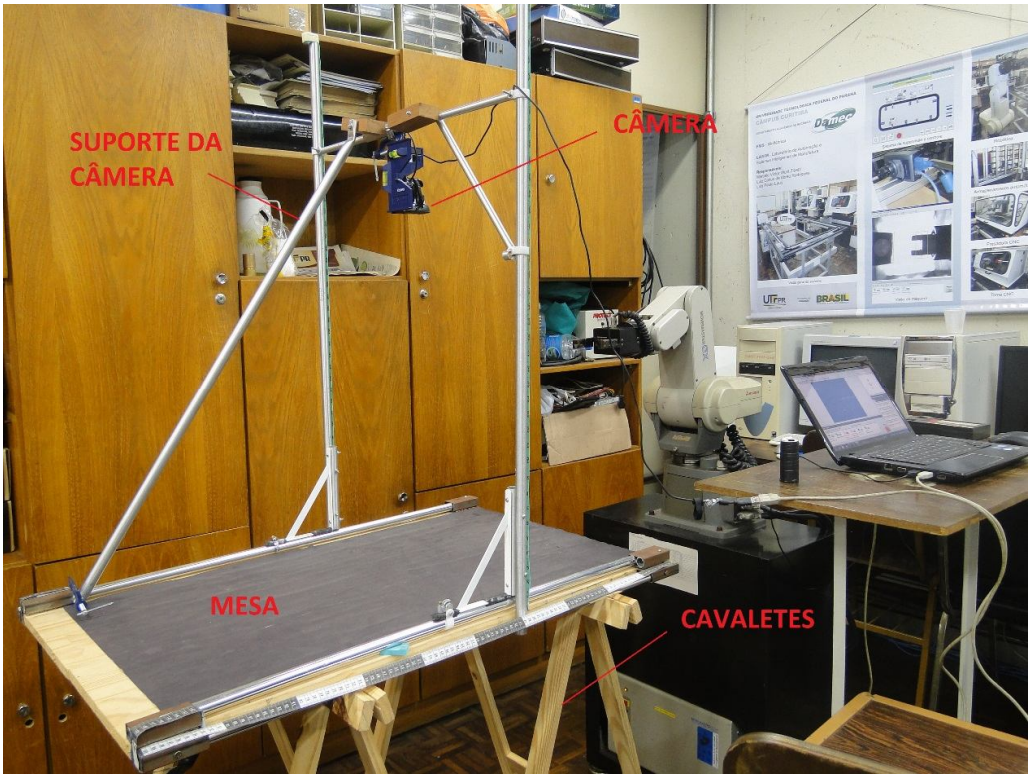


Figura 28: Mesa e suporte da câmera

Fonte: Autor

Acoplado à mesa, foi construído um suporte para a câmera, ilustrado também na Figura 30, feito com tubos de alumínio. Este suporte está paralelo à mesa devido a necessidade da câmera estar paralela à mesa, como definido na seção 3.1. Tal necessidade exigiu também que a câmera fosse acoplada em um medidor de nível, como é mostrado nas imagens da Figura 29. Dessa forma, garante-se, através dos indicadores de nível, que a angulação da câmera em relação aos eixos X e Y ao plano que a mesma observa, ou seja, a mesa, é sempre igual a 0°. A altura da mesa foi atribuída tendo como referência a altura da base do robô, de forma que seja suficiente para manipular a peça sobre a mesa.

Figura 29: Medidor de nível acoplado à câmera

Fonte: Autor

3.2.3 Corpo de prova

Foi utilizada uma peça cilíndrica composta por latão, usinada no próprio laboratório de FMS da UTFPR, com 33 mm de diâmetro e 73 mm de altura, como ilustra a Figura 30.

Para facilitar a identificação da peça durante o processamento de imagens e tendo em vista que a superfície da mesa é preta, a peça foi revestida inteiramente com fita preta e sua base cilíndrica foi revestida com papel branco.

Figura 30: Peça cilíndrica revestida

Fonte: Autor

3.2.4 Computador

Foi utilizado um *notebook Lenovo G460*, com processador *Intel Core* i5 M 430 2.27 GHz, 4 GB de RAM, placa de vídeo GeForce 310M com 512 MB de memória de vídeo dedicada e operando em Sistema Operacional Windows 7 Home Premium.

3.2.5 LabVIEW

Segundo Bovik (2009), o *software* da *National Instruments*, *LabVIEW* (*Laboratory Virtual Instrument Engineering Work-bench*), é um ambiente gráfico de desenvolvimento utilizado para criar variados tipos de aplicações destinadas para variados fins, como controle, estatística, monitoramento, entre outros. O *LabVIEW* é utilizado mundialmente em variadas áreas, tanto no meio industrial como no meio acadêmico.

A linguagem de programação do *LabVIEW* é fundamentada totalmente em um ambiente gráfico. Ao invés de se utilizar a clássica programação por texto, utilizam-se componentes visuais que se conectam entre si de acordo com suas respectivas funções. Assim, toda a lógica de programação, assim como o fluxo dos dados do programa é determinada de acordo com a forma com que esses componentes trocam informações entre si. É possível estruturar o programa através de vários blocos, onde cada um realiza uma determinada tarefa. Cada tarefa pode ser executada de maneira simultânea com outras tarefas, o que caracteriza o modo multitarefa de execução. Com isso, a implementação é simples e várias operações são executadas ao mesmo tempo.

No *LabVIEW* está disponibilizada uma enorme gama de ferramentas e funcionalidades que possibilita ao desenvolvedor, por exemplo, realizar avançados cálculos matemáticos; medição de sinais; criação de animações; construir gráficos com os dados adquiridos; monitorar processos em tempo real; compartilhar, adquirir e apresentar dados nos mais variados tipos de dispositivos; e, finalmente, realizar aquisição e processamento digital de imagens, esta última que foi profundamente abordada durante todo o desenvolvimento deste trabalho.

3.2.6 Vision Builder

National Instruments Vision Builder for Automated Inspection é um *software* interativo de ambiente configurável para desenvolver sistemas de visão artificial completos. Fornece soluções para a maioria dos processos com máquinas que

necessitam de aplicação de visão sem as complexidades da programação clássica por texto, conforme National Instruments (2013).

O *Vision Builder* é utilizado para processar imagens, identificar características em objetos, adquirir medidas, verificar a presença de objetos, dentre várias outras utilidades. *Softwares* de visão configuráveis, como o *Vision Builder*, normalmente usam uma seqüência linear de eventos para processar imagens, de acordo com Nicholas Vazquez, engenheiro chefe de *software* do *National Intruments Vision R & D Group*, conforme National Instruments (2013).

O *Vision Builder* permite configurar, medir o desempenho e implementar aplicações de visão completa da máquina. O usuário estabelece um fluxo de execução para o sistema de visão e a partir do alinhamento de componentes para inspeção e verificação das imagens adquiridas, estabelece-se se a inspeção foi aprovada ou não.

3.3 Iluminação

Após a aquisição de algumas imagens da câmera sobre a área de trabalho do braço robótico, optou-se por utilizar o sistema de iluminação direta. A iluminação do laboratório é satisfatória para tal sistema, isto é, gerando poucas sombras e sem qualquer tipo de interferência (excesso de luz por exemplo). As luminárias fornecem iluminação diretamente sobre a área de trabalho e com uma luminosidade suficiente.

3.4 Programa de processamento de imagens

A função do programa de processamento de imagens é definida pelos itens a seguir:

- Adquirir a imagem da câmera;
- Processar a imagem, de forma que o objeto a ser manipulado seja identificado, com seu formato bem definido. Neste trabalho o objeto em questão é uma peça cilíndrica, após o processamento da imagem adquirida, deverá ser gerada uma nova imagem com apenas uma forma circular (base cilíndrica) contida nela;
- Fornecer as coordenadas X e Y da peça no sistema cartesiano da câmera

Inicialmente, tentou-se desenvolver o programa de processamento de imagens inteiramente dentro do *software LabVIEW*. Porém, foi constatado que o tempo de duração para o desenvolvimento do programa seria muito longo. O tempo

para aprender a aplicar corretamente todas as técnicas de processamento de imagens dentro do *LabVIEW* interferiria no cronograma de execução do projeto, tendo em vista o conhecimento básico do *software* por parte do desenvolvedor deste projeto.

Após analisar os produtos e ferramentas disponibilizadas pela *National Instruments* para processamento de imagens, foi considerada a melhor opção o *software Vision Builder*. O contato prévio com o *software* durante o curso de Tecnologia em Mecatrônica Industrial da UTFPR, campus Curitiba, na disciplina de Projeto Integrador 3, foi de extrema importância. Durante a disciplina foi exigido que se desenvolvesse uma inspeção das dimensões das peças produzidas na linha de manufatura do laboratório de FMS. Pelo fato das peças analisadas serem todas cilíndricas, o programa desenvolvido na ocasião auxiliou no desenvolvimento do programa deste trabalho, em virtude de utilizar conceitos de processamento de imagens parecidos, como por exemplo, o reconhecimento de formas circulares na imagem adquirida. Outro motivo pela escolha da utilização do *Vision Builder* foi o fato deste *software* ser muito prático e fácil de usar. Em pouco tempo de desenvolvimento, em algumas horas por exemplo, pode-se desenvolver um programa que inspeciona imagens adquiridas por uma câmera. Considerando que a tarefa de processar imagens não é algo simples de se desenvolver, tendo como base, a complexidade dos cálculos realizados e as ações a serem tomadas pelo sistema, algumas horas é um tempo de desenvolvimento satisfatório. Também, o *Vision Builder* retira a necessidade de o desenvolvedor elaborar algoritmos com lógicas complexas para que o objetivo do funcionamento do programa seja alcançado. Já no *LabVIEW* isto não acontece, para que o processamento das imagens ocorra adequadamente, o desenvolvedor deve elaborar algoritmos complexos, combinando blocos e funções matemáticas, para obter o resultado esperado na execução do programa.

3.4.1 Etapas da inspeção

As etapas da inspeção realizadas no programa de processamento de imagens desenvolvido no *software Vision Builder*, são ilustradas na Figura 31, indicadas pelas letras de A até F.

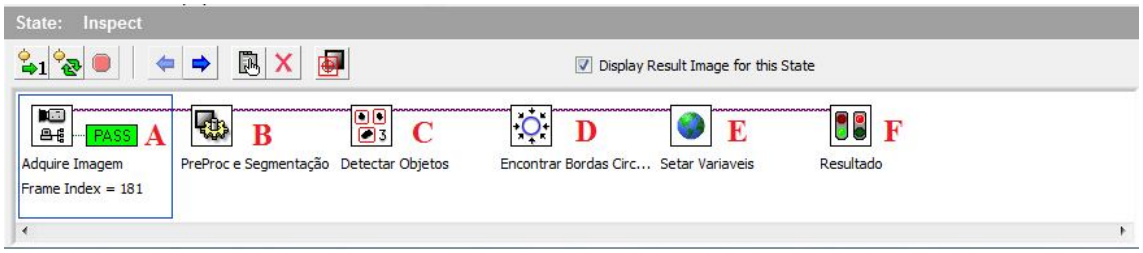


Figura 31: Etapas da inspeção realizada no *Vision Builder*

Fonte: Autor

A) Aquisição da imagem e calibração da câmera

Antes que qualquer tipo de processamento de imagem seja realizado no programa, primeiramente é necessário que a imagem seja adquirida da câmera. A primeira função desta etapa é capturar a imagem gerada pela *webcam Logitech HD C270*, continuamente e em tempo real.

Optou-se por adquirir uma imagem na maior resolução suportada pela *webcam* (1280 x 960 *pixels*) para que assim sejam geradas imagens com a maior qualidade possível, o que facilita a execução das posteriores etapas da inspeção. É verdade que uma imagem em resolução alta resulta em uma quantidade maior de informações, as quais podem aumentar o custo computacional durante o processamento. Porém, durante os testes realizados não foram constatadas grandes diferenças de velocidade de processamento utilizando imagens com resoluções menores. Uma imagem típica adquirida pela *webcam* é ilustrada na Figura 37.

A segunda função desta etapa é calibrar a câmera. O processo de calibração tem como objetivo aproximar ao máximo as dimensões lidas pelo *software Vision Builder* com as dimensões do mundo real. Ou seja, se 1280 x 960 *pixels* da imagem adquirida pela *webcam* representarem, por exemplo, uma área de 40 x 30 cm, então o programa deve representar essa área com os valores mais próximos possíveis. Para calibrar a câmera, é necessário executar os passos apresentados no apêndice L. Configuração utilizada: calibração simples, *pixels* quadrados e origem do sistema de coordenadas no centro da imagem adquirida.

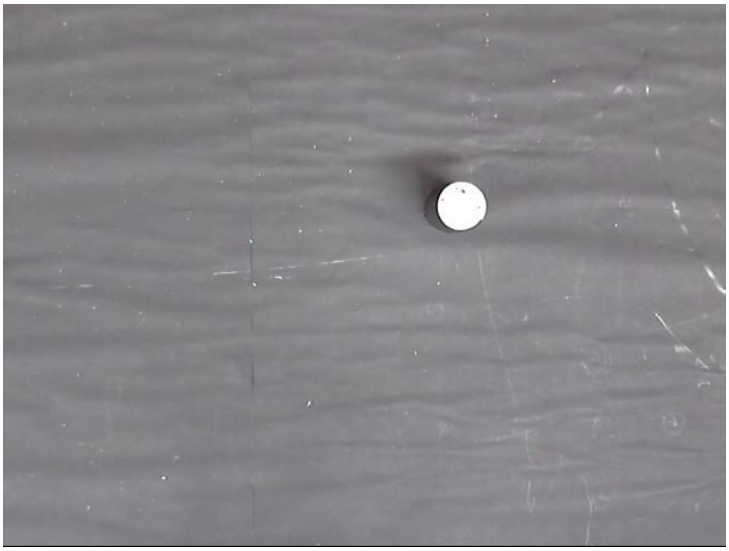


Figura 32: Imagem adquirida pela *webcam Logitech HD C270*

Fonte: Autor

B) Pré-processamento e segmentação

Nesta etapa é realizado o pré-processamento e a segmentação da imagem adquirida. Como explicado na seção 2.3 deste trabalho, o pré-processamento objetiva melhorar a qualidade da imagem de forma a diminuir a complexidade das etapas posteriores. Neste caso, o objetivo é destacar as formas e a localização da peça cilíndrica. Já na segmentação o objetivo é separar o que é de interesse do restante da imagem, ou seja, separar a peça cilíndrica do fundo da imagem.

Inicialmente é dado um nome para a etapa e posteriormente escolhe-se a região de interesse durante a execução da etapa. Escolheu-se a imagem inteira, já que a peça cilíndrica pode estar em qualquer lugar da imagem.

Para criar esta etapa são configuradas sub-etapas, como mostra a Figura 33, indicadas pelos números de 1 até 4.

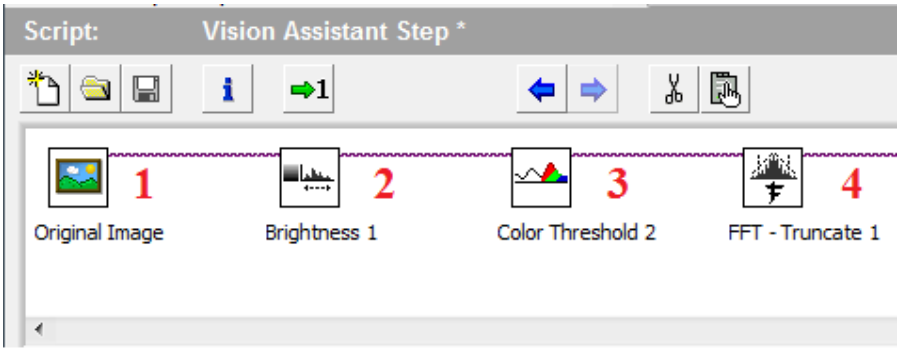


Figura 33: Sub-etapas da etapa de pré-processamento e segmentação

Fonte: Autor

1) **Imagem original:** o programa extrai a imagem previamente adquirida na etapa de aquisição de imagem. Nesta etapa, nenhum tipo de processamento é realizado, como mostra a Figura 34.

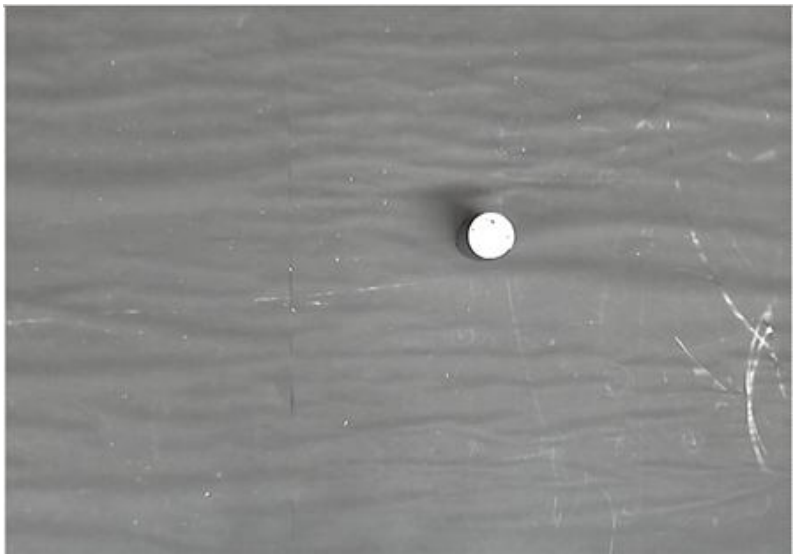


Figura 34: Imagem original

Fonte: Autor

2) **Manipulação de contraste e brilho:** ilustrada pela Figura 35, esta etapa aplica ajustes nos níveis de brilho, contraste e gama da imagem. O objetivo é destacar ao máximo a peça cilíndrica da superfície em que esta se situa, neste caso, a superfície da mesa.

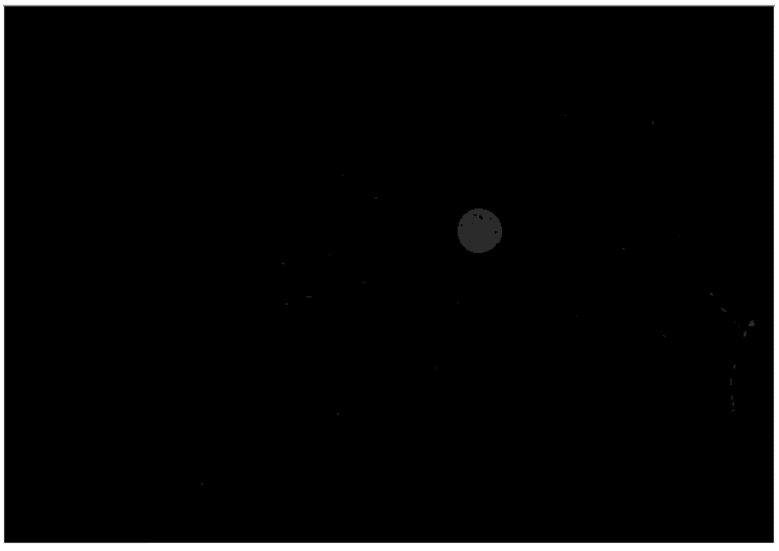


Figura 35: Ajuste de brilho

Fonte: Autor

3) **Limiarização:** nesta etapa é realizada uma detecção de limiares. Como explicado na seção 2.3.9, esta técnica procura dividir a imagem quando esta apresenta duas classes, o fundo e o objeto de interesse, no caso deste trabalho, a superfície da mesa e a peça cilíndrica respectivamente. A operação de limiarização divide porções da imagem tendo um nível de cinza como limite. Se o valor estiver abaixo do limite, então o nível zero é atribuído ao *pixel*, já se o valor estiver acima do limite, então o nível um é atribuído ao *pixel*. O limite utilizado foi o nível de cinza com valor igual a 33 (este valor pode variar dependendo dos ajustes realizados na etapa 2). A Figura 36 mostra o resultado desta operação.

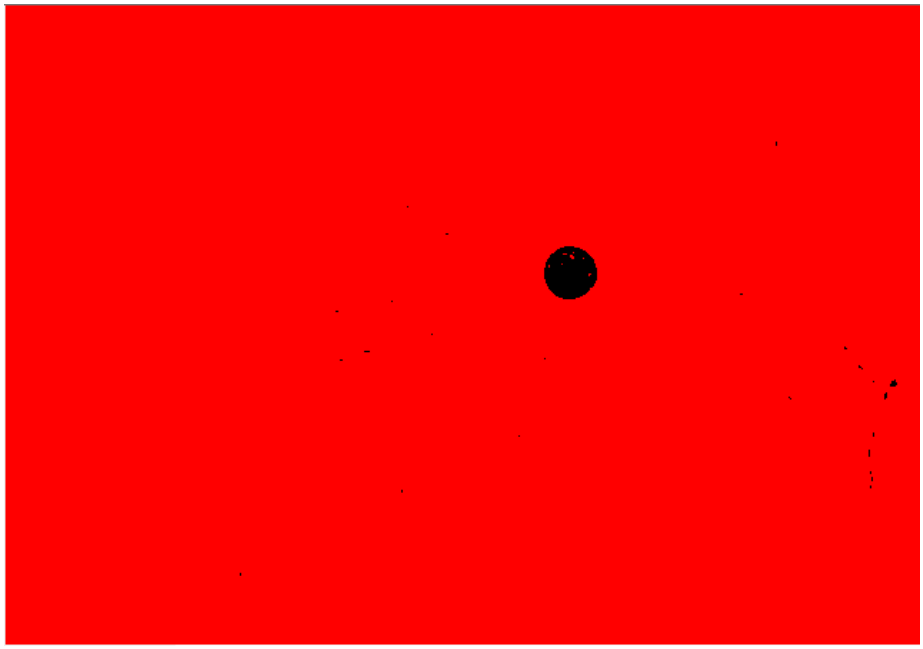


Figura 36: Limiarização da imagem

Fonte: Autor

Para ilustrar a operação foi escolhida a cor vermelha, apenas por ser uma cor que se diferencia bastante da cor preta. Todavia, qualquer outra cor poderia ter sido escolhida.

4) **Filtro de frequência:** percebe-se que a imagem resultante da etapa de limiarização (Figura 36) traz um resultado consideravelmente satisfatório para o objetivo do processamento, separando bem a peça cilíndrica do restante da

imagem. Porém, observa-se que esta imagem apresenta alguns *pixels* ruidosos, como destaca a Figura 37.

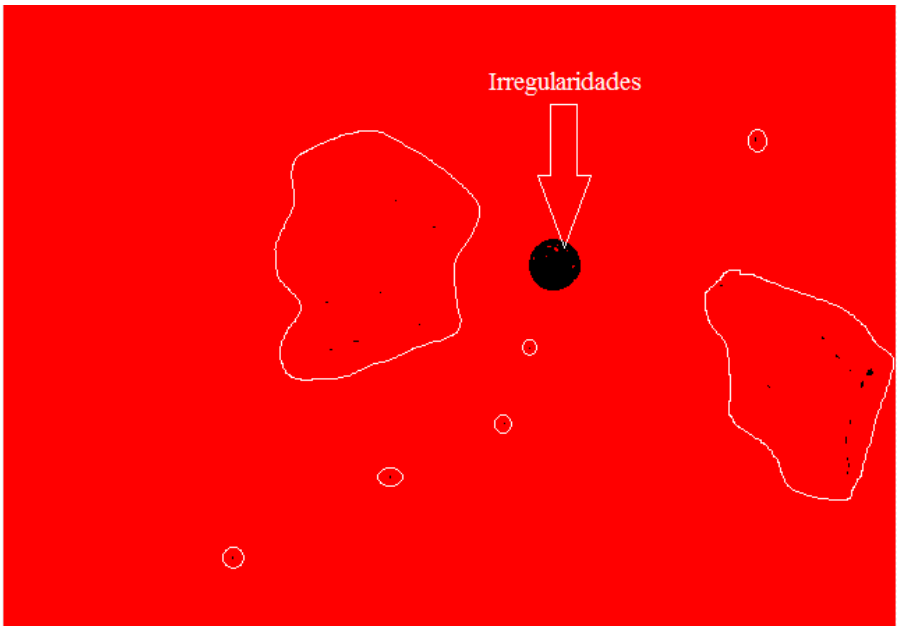


Figura 37: Ruídos na imagem após a limiarização

Fonte: Autor

Para corrigir tais ruídos, optou-se por utilizar a técnica de filtro de frequência. A seção 2.3.6, explica que esta técnica objetiva alterar valores de *pixels* de acordo com a periodicidade e distribuição espacial das variações de intensidade luminosa na imagem. Optou-se por utilizar o filtro de freqüência passa-baixa ideal, também descrito na seção 2.3.6, utilizado para atenuar ou remover, ou truncar, altas frequências presentes na imagem. Este filtro suprime informações relacionadas com rápidas variações de intensidades luminosas na imagem espacial. A FFT inversa, utilizada depois do filtro de freqüência passa-baixa truncado, produz uma imagem onde ruídos, detalhes, textura e arestas são suavizadas ao máximo. Desta forma, os ruídos da imagem da Figura 37 são eliminados e o formato da peça cilíndrica é mais bem definido, como mostra a Figura 38.

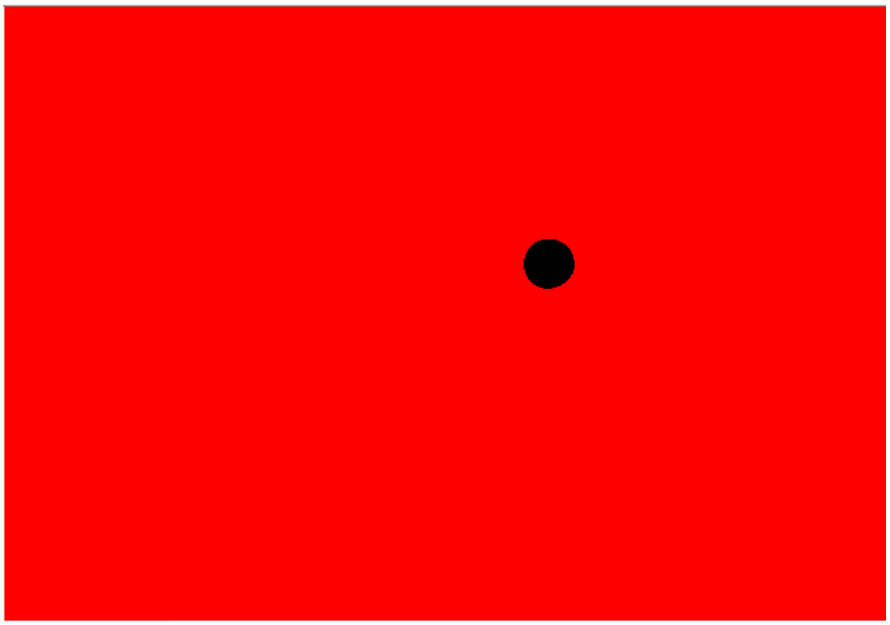


Figura 38: Filtro de frequência aplicado na imagem

Fonte: Autor

C) Detecção do objeto

Nesta etapa é realizada a detecção do objeto na imagem e se nela existe realmente apenas **um** objeto.

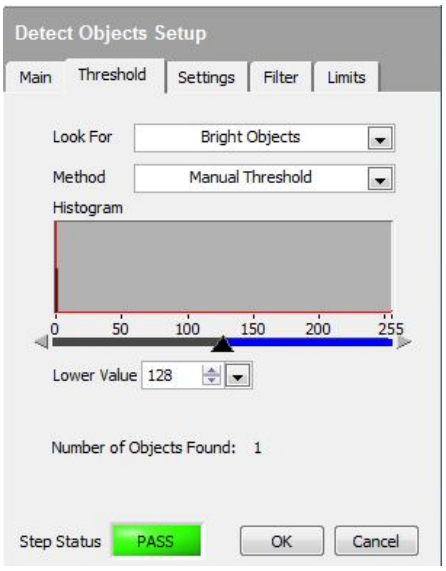


Figura 39: Limiarização da etapa de detecção de objetos

Fonte: Autor

Sendo assim, a técnica de limiarização, anteriormente utilizada na etapa de pré-processamento e segmentação da imagem, novamente é utilizada na etapa de detecção de objetos como ilustra a Figura 39. Para encontrar objetos são analisadas mudanças dos níveis de cinza da imagem e estabelece-se uma procura por objetos brilhantes pelo método de limiarização manual. Este tipo de limiarização foi escolhida por exigir menor capacidade computacional e por as imagens processadas neste trabalho serem simples, isto é, com pouca informação. Com isso, o *software* constrói um histograma que mostra as variações dos níveis de cinza. Para que os objetos sejam identificados através de tais variações, é necessário determinar uma faixa de valores entre 0 e 255 na escala de cinza, onde o nível 0 é caracterizado pela cor preta e o 255 pela cor branca. Os objetos que possuírem níveis de cinza dentro da faixa de valores pré-determinada serão considerados nesta etapa. A fim de identificar os objetos brilhantes, a faixa determinada foi entre 128 e 255, como mostra a Figura 40. Utilizou-se esta faixa por possuir os valores que geralmente objetos brilhantes (entre 128 e 191) e muito brilhosos (entre 191 e 255) possuem.

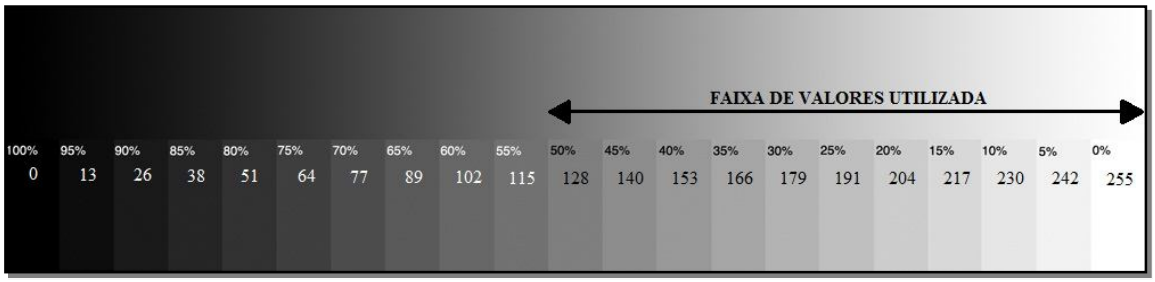


Figura 40: Faixa de valores utilizada na escala de cinza

Fonte: Adaptado de PPAINTINGA

Para garantir que após esta etapa exista apenas um objeto na imagem, deve-se configurar os valores de número mínimo e máximo de objetos para **um**. Já que o sistema de visão foi desenvolvido para apenas **uma** peça.

Em seguida é necessário estabelecer um tamanho máximo do objeto que será encontrado na imagem. No *software* o tamanho é determinado pela área ocupada pelo objeto na imagem. Após analisar experimentalmente vários casos com a peça cilíndrica, constatou-se que o tamanho máximo da área do objeto a ser aceita por esta etapa é de 6.400 mm². A peça a ser manipulada não pode ter dimensões que ultrapassam os limites do efetuador do braço robótico, a garra. Caso contrário, a peça pode ser maior que as dimensões da garra do robô, tornando a tarefa de manipulação impossível. Se não houver essa condição, o robô irá realizar esforços

desnecessários que podem causar danos em sua estrutura. Tendo em vista que o atuador possui dimensões 100 mm x 80 mm, então pode-se supor (utilizando uma tolerância de 10 mm) que o mesmo pode carregar uma peça cilíndrica de diâmetro máximo igual a 90 mm, cuja área circular será no máximo aproximadamente igual a 6.400 mm².

O resultado desta etapa é ilustrado na Figura 41, onde se observa a identificação de apenas um objeto na imagem.

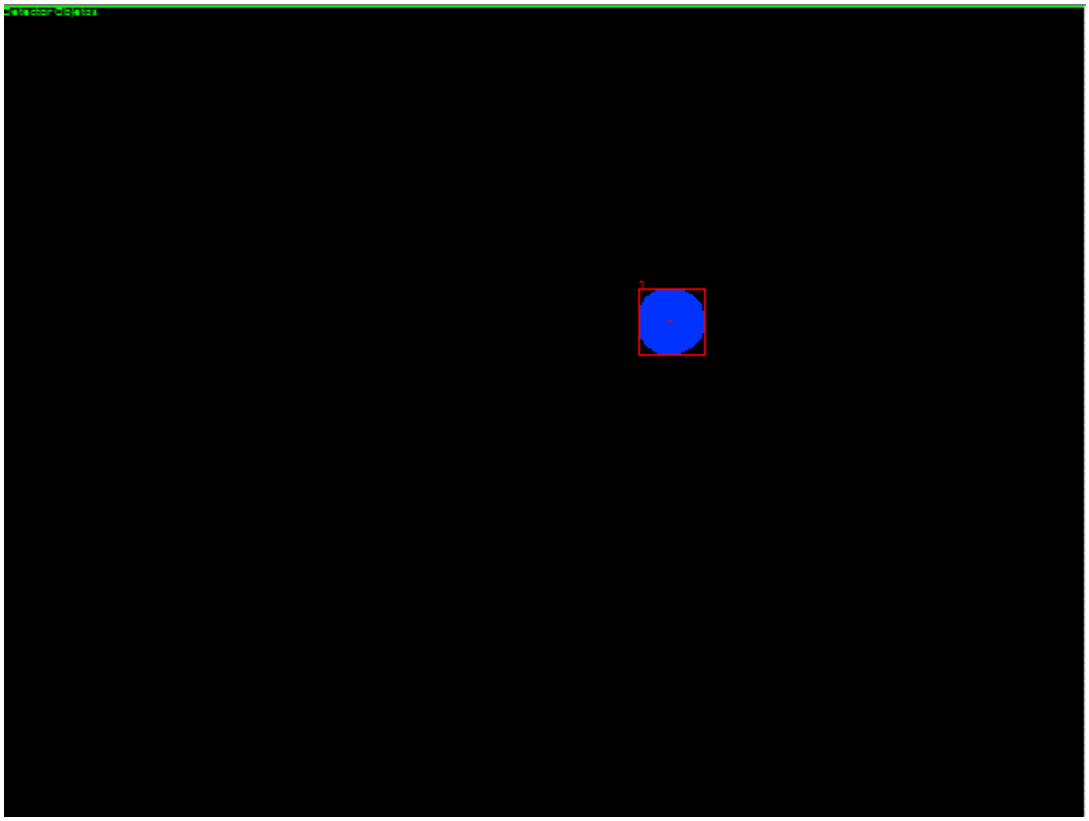


Figura 41: Peça cilíndrica identificada pela etapa de detecção de objetos

Fonte: Autor

D) Identificação de bordas circulares

Após o sistema detectar a existência de apenas um objeto na imagem, o próximo passo é garantir que tal objeto possui um formato circular. Para tanto, foi utilizada a técnica de segmentação por detecção de bordas relatada na seção 2.3.8 deste trabalho. Esta técnica analisa regiões na imagem com mudanças abruptas do nível de intensidade luminosa. Neste trabalho foi utilizada a descontinuidade do tipo borda e analisou-se se as bordas identificadas resultam em uma forma circular.

O resultado desta etapa é ilustrado pela Figura 42, onde se observa que o formato circular é destacado.

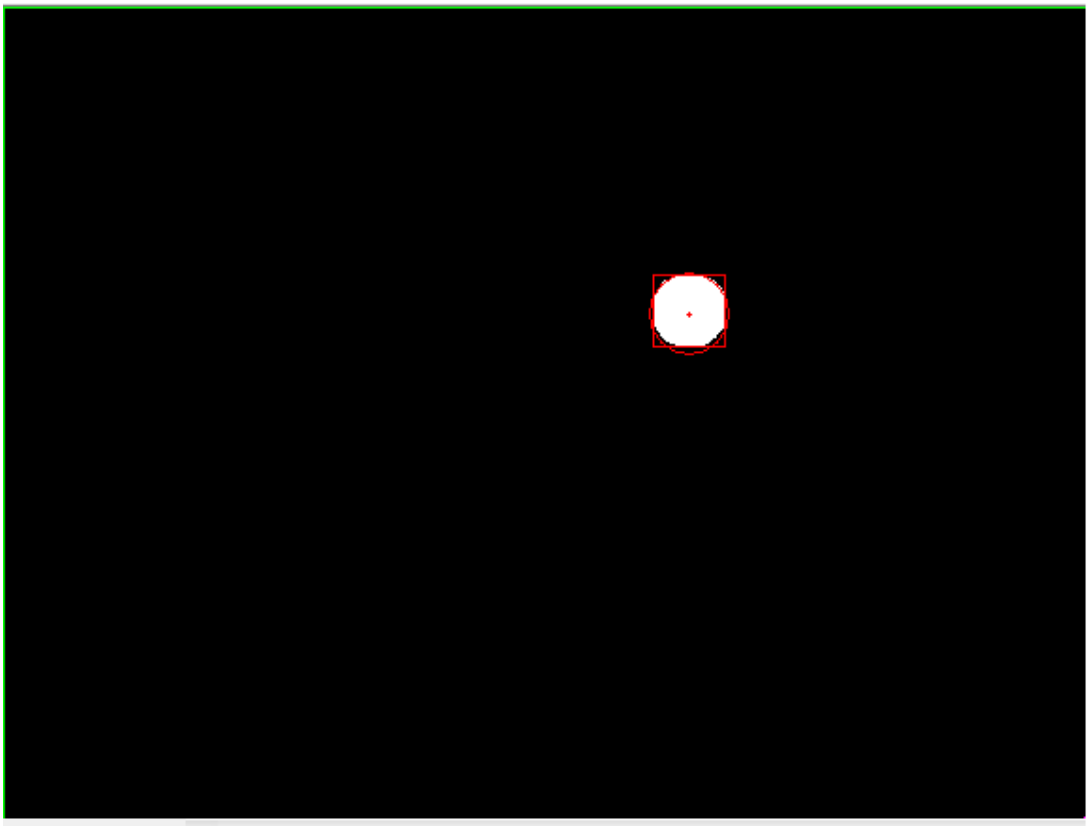


Figura 42: Borda circular identificada pela etapa de identificação de bordas circulares

Fonte: Autor

E) Definição dos valores das variáveis

Para que as informações geradas pelo programa de processamento de imagens (desenvolvido no *Vision Builder*) sejam utilizadas pelo programa de manipulação (desenvolvido no *LabVIEW*), é necessário que tais informações sejam enviadas constantemente em tempo-real do *software Vision Builder* para o *LabVIEW*. Com isso, foi utilizado o gerenciador de variáveis disponível no *software Vision Builder*, como mostra a Figura 43.

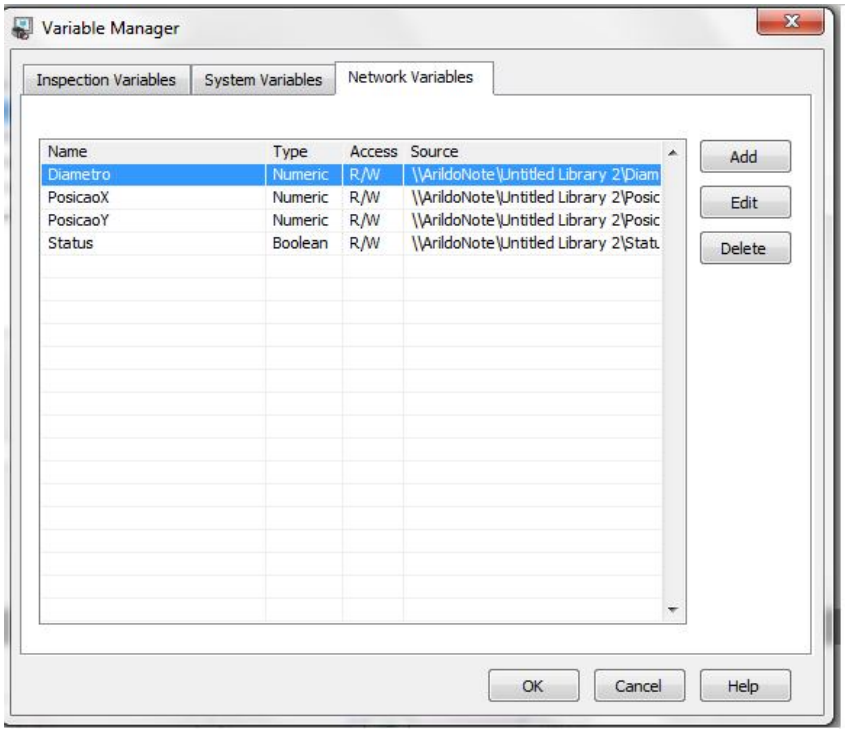


Figura 43: Gerenciador de variáveis do *software Vision Builder*

Fonte: Autor

Foram criadas quatro variáveis que são constantemente lidas pelo programa de manipulação: o diâmetro da peça, as coordenadas X e Y do centro da peça cilíndrica (todos encontrados na execução da etapa de detecção de bordas circulares) e o estado final da inspeção que é o resultado da execução da última etapa do programa de processamento de imagens.

Como tanto o programa de processamento de imagens como o programa de manipulação estão sendo executados no mesmo computador, não há necessidade de criar uma rede local ou haver acesso à internet para que haja comunicação entre os dois programas. Portanto, a comunicação entre o *Vision Builder* e o *LabVIEW* é realizada através do acesso às variáveis localizadas no sistema de arquivos existente no computador.

Para cada variável criada é associada uma medida adquirida em uma das etapas da inspeção. Por exemplo, no caso da variável “Posicao X” foi associada a medida da coordenada X do centro da peça na etapa **D** de detecção de bordas circulares.

F) Estado final da inspeção

Esta etapa apenas verifica constantemente se todas as etapas anteriores foram aprovadas. Se uma das etapas anteriores falhar então o estado final da inspeção da imagem será de “FALHA” (*FAIL*), caso contrário será de “APROVADO” (*PASS*), como mostra a Figura 44.

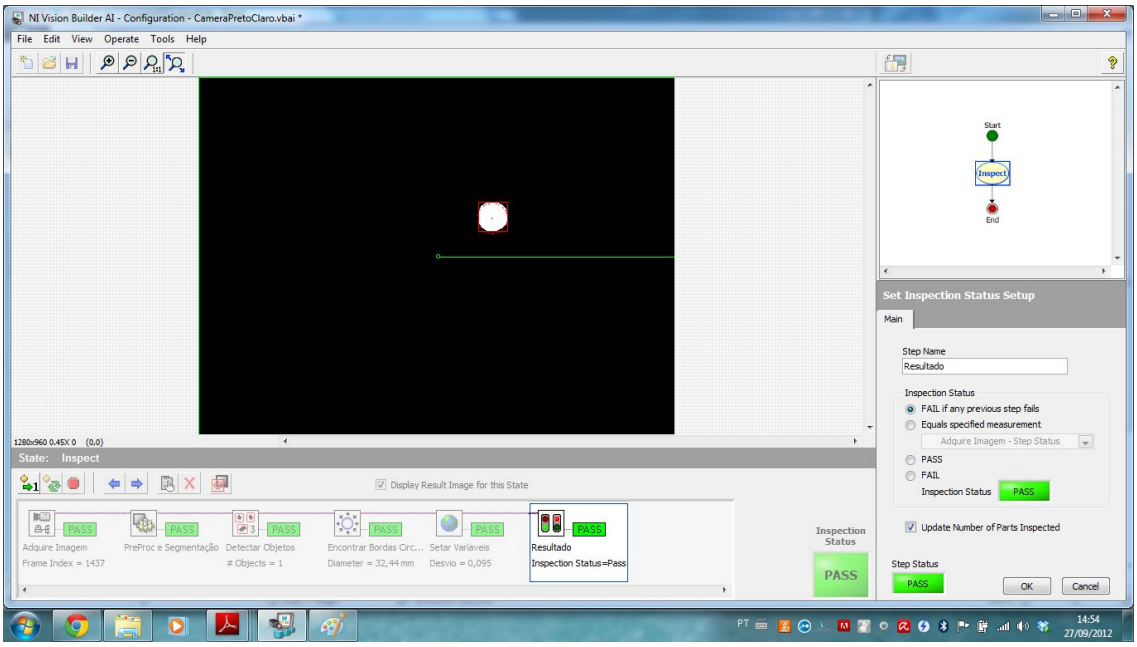


Figura 44: Estado final da inspeção

Fonte: Autor

3.5 Programa de manipulação

A função do programa de manipulação é definida pelos itens a seguir:

- Ler e processar constantemente as informações geradas pelo programa de processamento de imagens que identifica a peça a ser manipulada;
- Realizar os cálculos necessários, de forma que instruções sejam geradas corretamente para o robô manipular a peça cilíndrica em qualquer lugar a seu alcance dentro do campo de visão da câmera;
- Supervisionar todo o sistema, monitorando o sistema de visão, as ações do robô e as medidas que são adquiridas. Isto é, atuar como um sistema supervisório.

Desde o início do desenvolvimento deste trabalho optou-se por utilizar o *LabVIEW* para desenvolver o programa de manipulação. Devido à possibilidade de estruturar todo o programa através de blocos, a velocidade de desenvolvimento é

aumentada, facilitando a construção de algoritmos e elaboração de lógicas. Também suas funcionalidades e ferramentas possibilitam executar cálculos matemáticos, controlar variáveis e monitorar processos em tempo real de maneira muito simples. A capacidade do *LabVIEW* se comunicar facilmente com o *software Vision Builder* também foi determinante na escolha do mesmo para desenvolver o sistema.

O fluxo de execução do programa de manipulação é ilustrado a seguir:

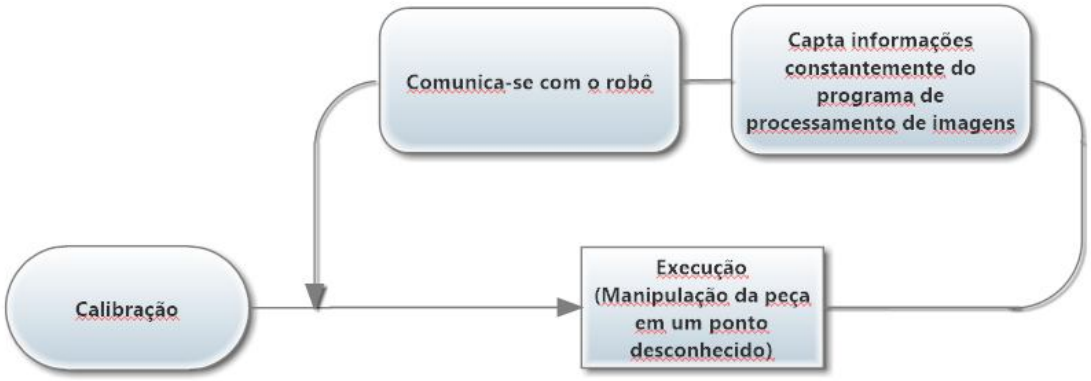


Figura 45: Fluxo de execução do programa de manipulação

Fonte: Autor

3.5.1 Comunicação com o programa de processamento de imagens

Antes de executar as etapas de calibração e execução, é necessário que o programa de manipulação leia constantemente as informações geradas pelo programa de processamento de imagens. Para tanto, é necessário que o *LabVIEW* acesse os valores das variáveis lidas durante a inspeção das imagens a qual foi realizada no *software Vision Builder*. Para tanto, foi utilizada uma funcionalidade disponível dentro do *LabVIEW* chamada *Data Binding*, que segundo National Instruments (2013) habilita ao programa atribuir todos os valores gerados no *Vision Builder* serem atribuídos para cada variável correspondente no *LabVIEW*.

3.5.2 Calibração

Conforme a seção 2.4, o sistema de visão robótico deste trabalho trabalha com dois sistemas de coordenados distintos posicionados de forma que seus eixos *X* e *Y* não são paralelos e as origens estão em locais diferentes e desalinhadas. Com isso, são aplicados conceitos de topografia e geodésia para efetuar a rotação

dos sistemas para que os eixos *X* e *Y* fiquem paralelos e calcular a translação dos sistemas para que a origem dos mesmos seja compatibilizada. A Figura 46 ilustra a representação dos sistemas de coordenadas deste trabalho.

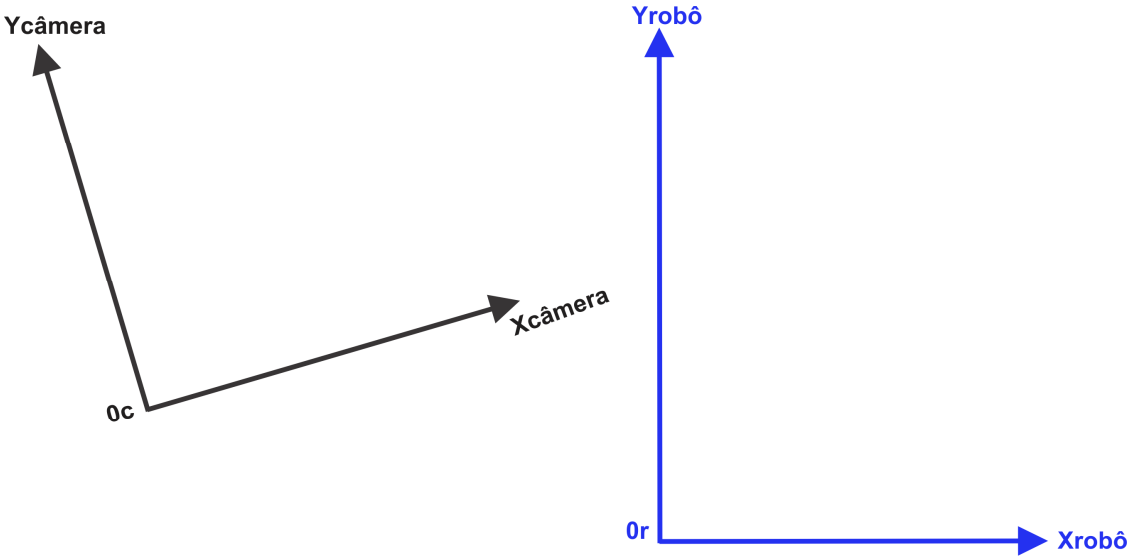


Figura 46: Representação dos sistemas de coordenadas deste trabalho

Fonte: Autor

Também é provável que haja erros nas medidas adquiridas pelo programa de manipulação, conforme citado na seção 2.4.4.

A etapa de calibração é responsável por calcular a rotação e translação entre os sistemas de coordenadas do robô e da câmera, assim como calcular os erros das medidas obtidas, para que então o sistema possa ser executado.

O fluxo de execução da etapa de calibração é ilustrado pela Figura 47 a seguir.

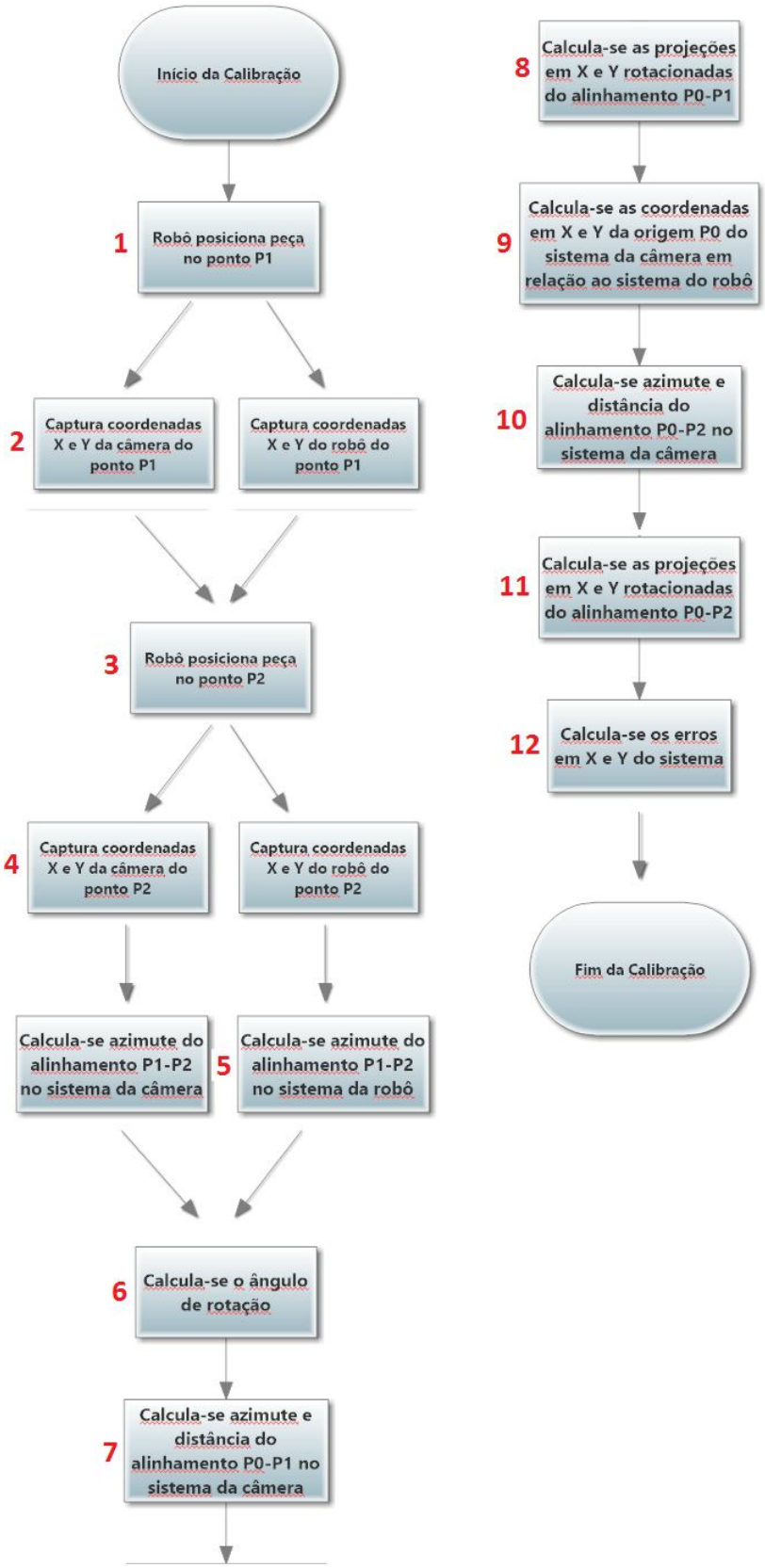


Figura 47: Fluxo de execução da etapa de calibração

Fonte: Autor

Para auxiliar o entendimento da etapa de calibração, a Figura 48 representa os sistemas de coordenadas deste trabalho na calibração. Os pontos P_1 e P_2 são pontos aleatórios pré-estabelecidos pelo operador do sistema para serem utilizados na calibração, isto é, são pontos cujas posições no espaço bidimensional foram pré-gravadas na memória da central do robô pelo operador do sistema. O sistema de coordenadas do robô foi considerado como de referência ("georreferenciado"), enquanto o sistema de coordenadas da câmera foi considerado como não georreferenciado. O ponto P_1 no sistema de coordenadas do robô foi chamado de Pg_1 e no sistema de coordenadas da câmera de P_1. Da mesma forma, o ponto P_2 foi nomeado.

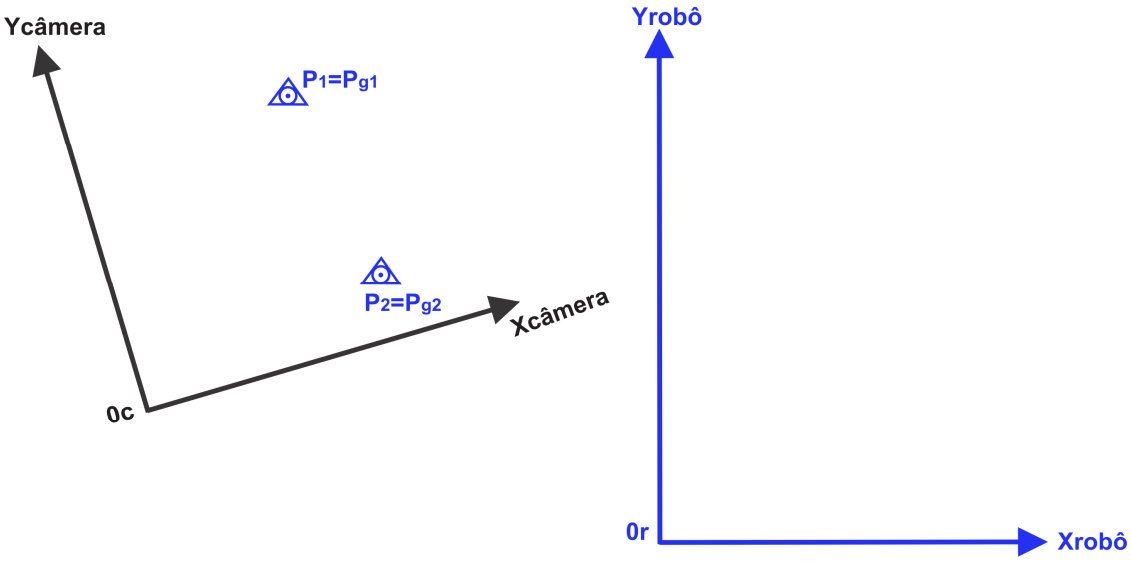


Figura 48: Representação dos sistemas de coordenadas deste trabalho na etapa de calibração

Fonte: Autor

Ao observar o fluxo de execução da Figura 47, nota-se que a etapa de calibração é dividida em 12 passos descritos a seguir:

1) **Robô posiciona a peça no ponto P_1:** é o posicionamento da peça cilíndrica na posição pré-gravada representada pelo ponto P_1 na Figura 48. Inicialmente, o programa de manipulação envia instruções para que o braço robótico pegue a peça em uma posição qualquer pré-gravada em sua memória (Figura 49), depois, o robô leva a peça até o ponto P_1 (Figura 50). Posteriormente, o robô retorna para uma posição de repouso pré-gravada

(Figura 51). Durante todo o processo, posições de aproximação são utilizadas a fim de evitar colisões durante a movimentação do robô.

Figura 49: Robô pega a peça em uma posição pré-gravada qualquer

Fonte: Autor

Figura 50: Robô leva a peça até o ponto P_1

Fonte: Autor

Figura 51: Robô retorna para a posição de repouso

Fonte: Autor

Figura 52: Robô leva a peça até o ponto P_2

Fonte: Autor

2) **Captura das coordenadas *X* e *Y* da câmera e do robô para o ponto P_1:** o programa de manipulação, sendo executado no *LabVIEW*, captura as coordenadas lidas pelo programa de processamento de imagens, sendo executado no *Vision Builder*, adquirindo assim as coordenadas do sistema da câmera (X_1, Y_1). Posteriormente, o programa captura as coordenadas pré-gravadas na memória do robô e assim adquire as coordenadas do sistema do robô (Xg_1, Yg_1).

3) **Robô posiciona a peça no ponto P_2:** o robô que estava na posição de repouso recebe instruções do programa para pegar a peça no ponto P_1. Após, a peça é manipulada até uma nova posição pré-gravada representada pelo ponto P_2 (Figura 52). Por fim, o robô retorna para sua posição de repouso (Figura 51).

4) **Captura das coordenadas *X* e *Y* da câmera e do robô para o ponto P_2:** são realizadas as mesmas operações do passo 2, porém para o ponto P_2. Com isto, os dados processados pelo programa de manipulação até então são ilustrados na Figura 53 abaixo.

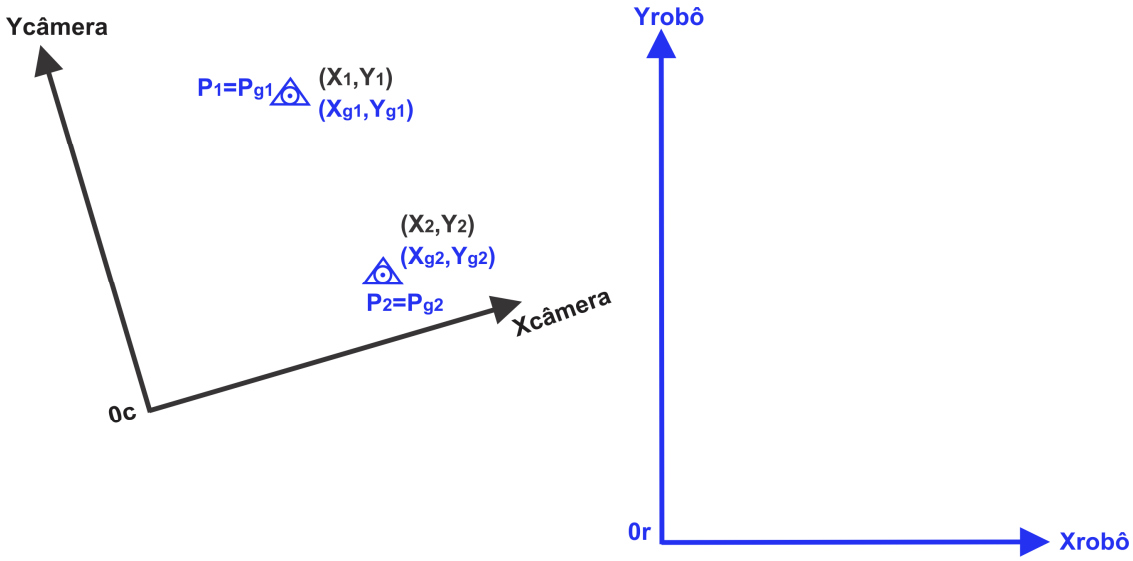


Figura 53: Dados processados pelo programa de manipulação até o passo 4 da etapa de calibração

Fonte: Autor

5) **Cálculo do azimute do alinhamento $P_1 - P_2$ no sistema da câmera e do robô:** são realizados os cálculos do azimute do alinhamento $P_1 - P_2$ (ilustrado na Figura 54) para ambos os sistemas de coordenadas da mesma forma apresentada na seção 2.4. As equações 49 e 50 representam o cálculo para o sistema de coordenadas da câmera (de forma análoga o mesmo cálculo foi feito para o sistema do robô).

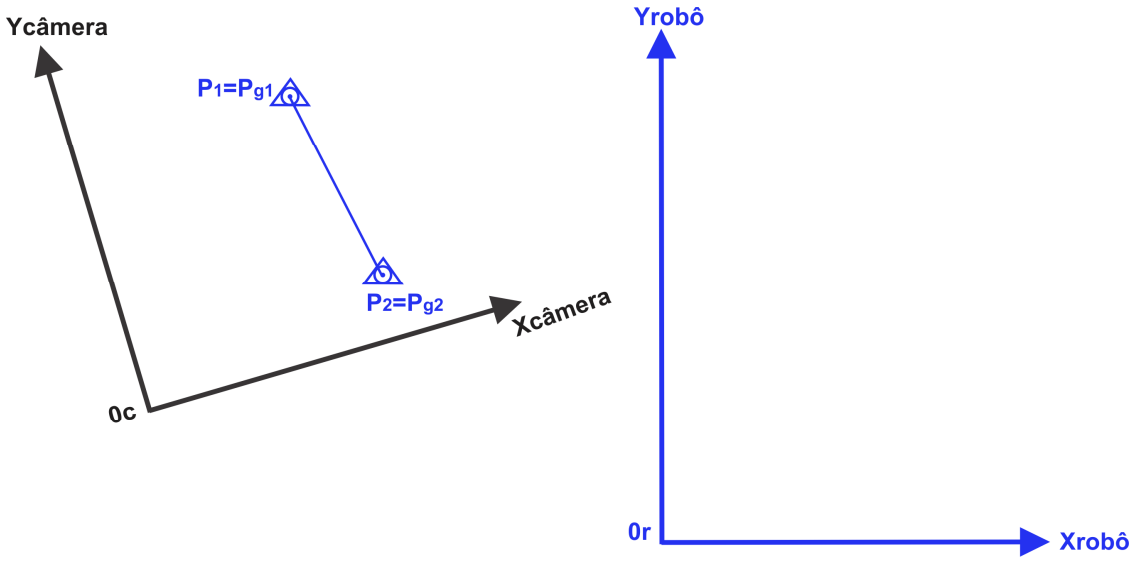


Figura 54: Alinhamento $P_1 - P_2$ na etapa de calibração

Fonte: Autor

$$\alpha_{1-2} = \arctan(\varphi) \tag{14}$$

Fonte: BORGES & CORDEIRO, (2012)

$$\tan\varphi = (X_2 - X_1) / (Y_2 - Y_1) \tag{15}$$

Fonte: BORGES & CORDEIRO, (2012)

Onde φ determina o azimute pelo conjunto de sinais, conforme tabela 3 abaixo:

$(X_2 - X_1) > 0$ e $(Y_2 - Y_1) > 0$	$\alpha_{1-2} = \varphi$
$(X_2 - X_1) > 0$ e $(Y_2 - Y_1) < 0$	$\alpha_{1-2} = 180 - \varphi$
$(X_2 - X_1) < 0$ e $(Y_2 - Y_1) < 0$	$\alpha_{1-2} = 180 + \varphi$
$(X_2 - X_1) < 0$ e $(Y_2 - Y_1) > 0$	$\alpha_{1-2} = 360 - \varphi$

Tabela 3: Combinação de sinais para determinação do azimute α_{1-2} (câmera e robô) na etapa de calibração

Fonte: BORGES & CORDEIRO, (2012)

Resultando nos azimutes da câmera (referenciado pelo norte magnético - Nm) α_{1-2} e do robô (referenciado pelo norte verdadeiro - Nv) β_{1-2} da Figura 55.

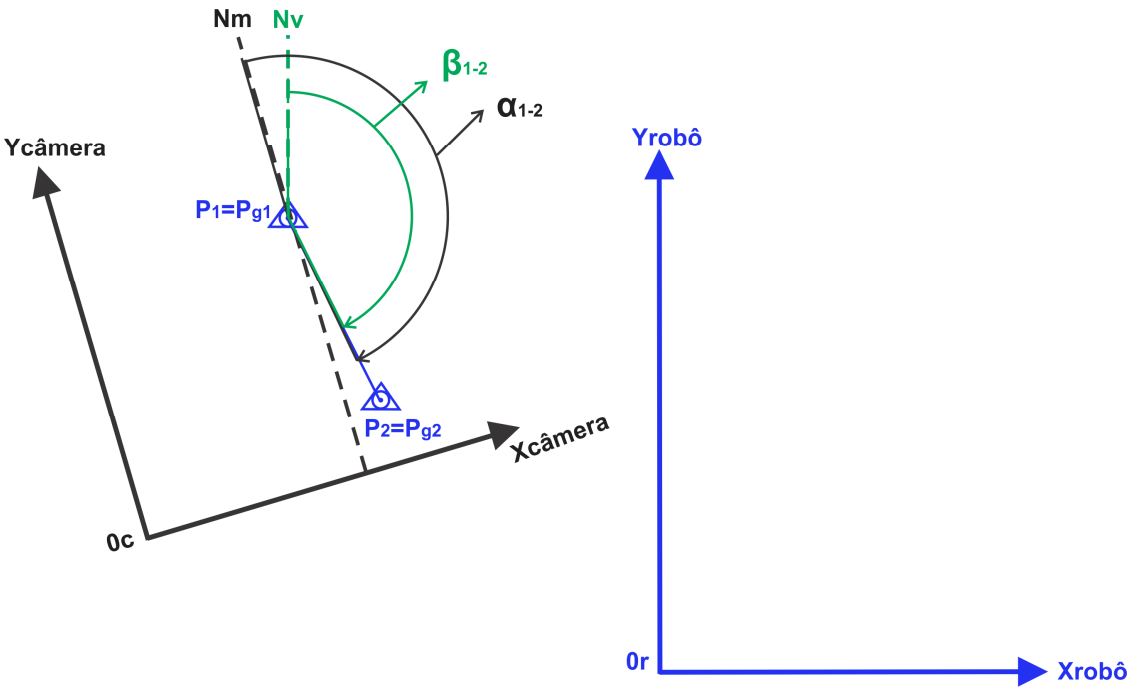


Figura 55: Azimutes do alinhamento $P_1 - P_2$ após realizado o passo 5 da etapa de calibração

Fonte: Autor

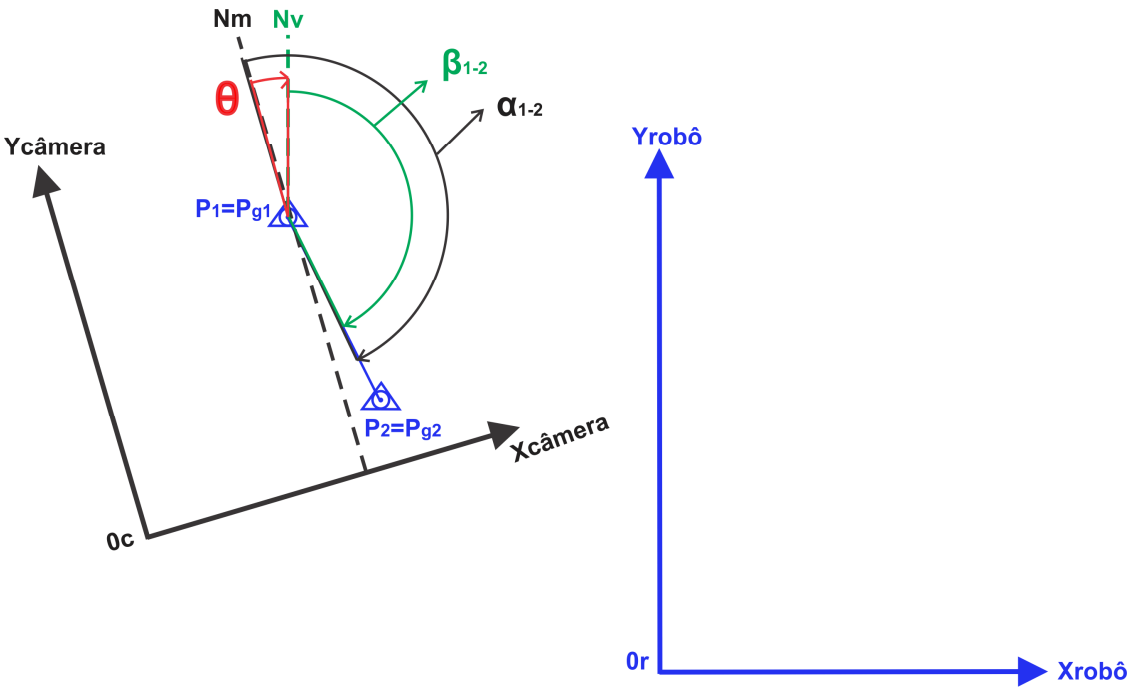


Figura 56: Ângulo de rotação θ calculado no passo 6 da etapa de calibração

Fonte: Autor

6) **Cálculo do ângulo de rotação:** é realizado o cálculo de rotação da mesma forma apresentada na seção 2.4.3, como mostra a equação 16 abaixo.

$$\theta = \alpha_{1-2} - \beta_{1-2} \tag{16}$$

Fonte: BORGES & CORDEIRO, (2012)

Obtém-se assim o ângulo de rotação θ ilustrado na Figura 56. Com isso, os eixos X e Y de ambos os sistemas de coordenadas estão paralelos. A partir do passo 7 começa o cálculo de translação dos sistemas para que a origem dos mesmos seja compatibilizada (onde obtém-se as coordenadas da origem do sistema da câmera no sistema do robô).

7) **Cálculo da distância e azimute do alinhamento $P_0 - P_1$ no sistema da câmera:** o objetivo é calcular as coordenadas da origem (P_0) do sistema da câmara no sistema do robô. Para tanto, primeiramente deve-se calcular o azimute do alinhamento $P_0 - P_1$ α_{0-1}, realizando os cálculos de forma similar aos apresentados no passo 5.

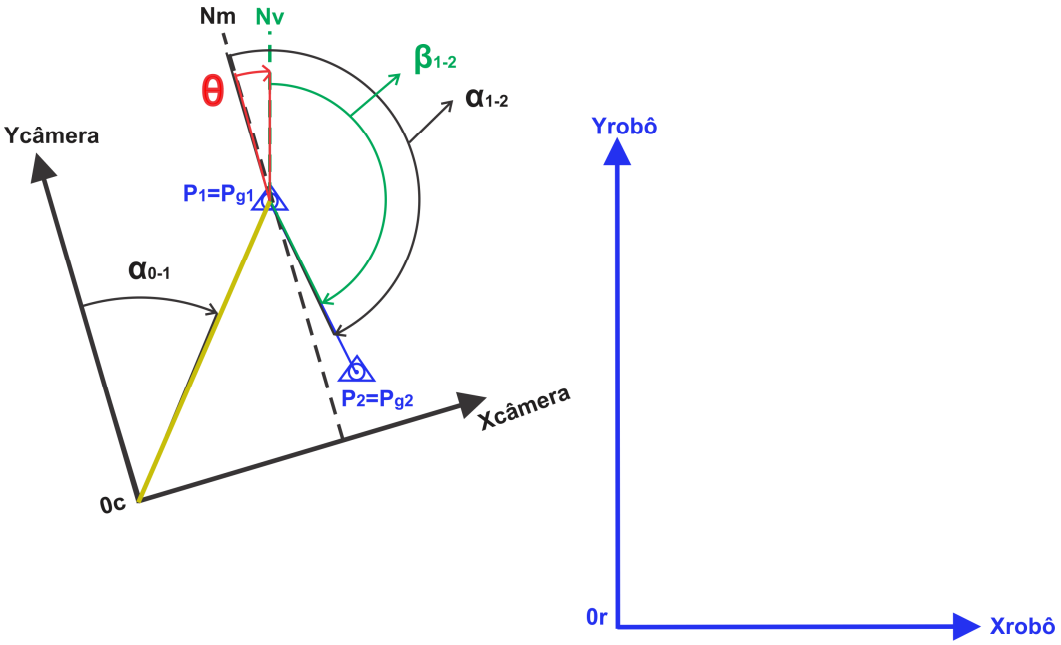


Figura 57: Azimute do alinhamento $P_0 - P_1$ α_{0-1} na etapa de calibração

Fonte: Autor

Feito isto, é necessário calcular a distância do alinhamento D_{O-1} entre os pontos P_0 e P_1 (ilustrado na Figura 58) conforme abaixo:

$$D_{O-1} = \sqrt{(X_1 - X_0)^2 + (Y_1 - Y_0)^2} \quad (17)$$

Fonte: BORGES & CORDEIRO, (2012)

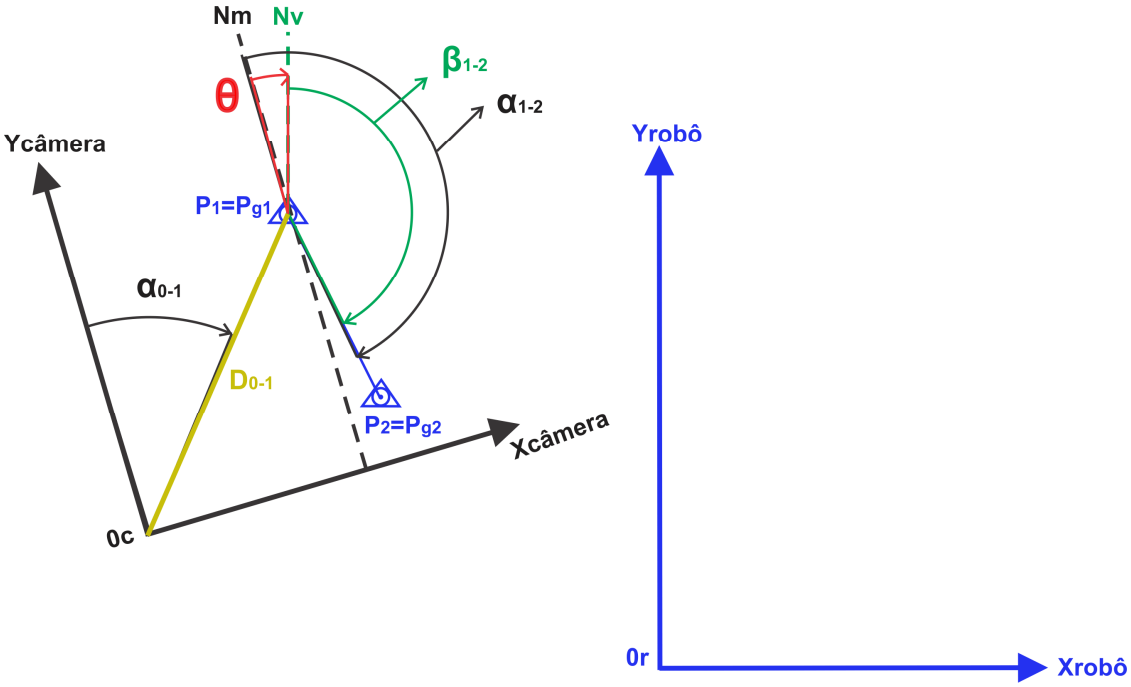


Figura 58: Distância D_{O-1} do alinhamento $P_0 - P_1$ na etapa de calibração

Fonte: Autor

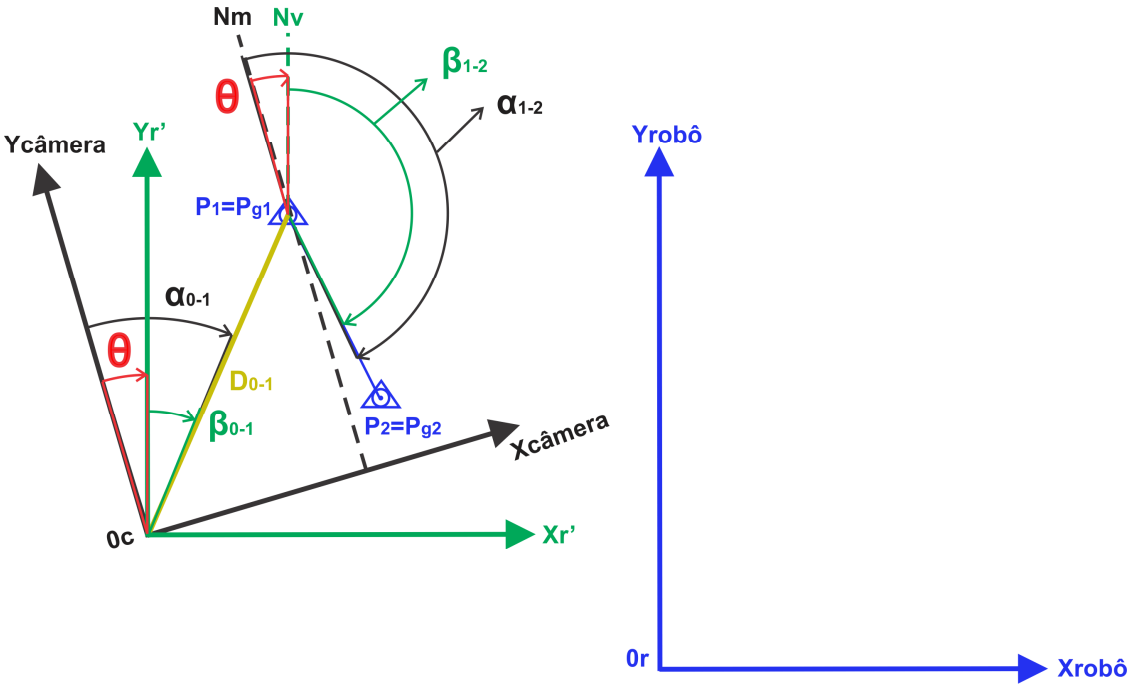


Figura 59: Azimute β_{0-1} do alinhamento $P_0 - P_1$ na etapa de calibração

Fonte: Autor

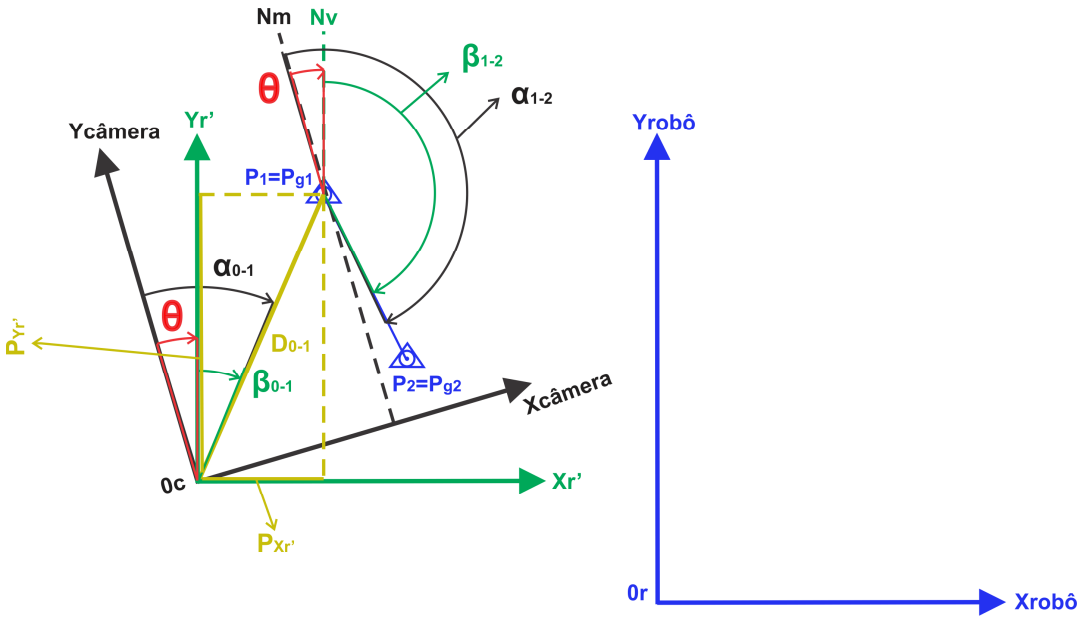


Figura 60: Projeções em X e Y rotacionadas do alinhamento $P_0 - P_1$ na etapa de calibração

Fonte: Autor

8) **Cálculo das projeções em *X* e *Y* rotacionadas do alinhamento $P_0 - P_1$:** segundo os cálculos da seção 2.4, primeiramente é necessário realizar o cálculo do azimute no sistema do robô do alinhamento $P_0 - P_1$ para que as projeções rotacionadas $P_{Xr'}$ e $P_{Yr'}$ possam ser calculadas. A sequência de cálculos é descrita a seguir.

Com o α_{0-1} calculado no passo 7, pode-se calcular o β_{0-1} (ilustrado na Figura 59) da seguinte forma (onde θ é a rotação calculada anteriormente):

$$\beta_{0-1} = \alpha_{0-1} - \theta \quad (18)$$

Fonte: BORGES & CORDEIRO, (2012)

Observa-se na Figura 59, um sistema de coordenadas $(X_{r'}, Y_{r'})$ destacado na cor verde que é o resultado da rotação do sistema da câmera para o sistema do robô. Com este azimute β_{0-1} e a distância D_{O-1} (calculada no passo 7) calcula-se as projeções em X e em Y $(P_{Xr'}, P_{Yr'})$, ilustradas na Figura 60) do alinhamento $P_0 - P_1$ no sistema verdadeiro, da seguinte forma:

$$P_{Xr'} = D_{O-1}\,\mathrm{sen}(\beta_{0-1}) \quad (19)$$

Fonte: BORGES & CORDEIRO, (2012)

$$P_{Yr'} = D_{O-1}\cos(\beta_{0-1}) \quad (20)$$

Fonte: BORGES & CORDEIRO, (2012)

9) **Cálculo das coordenadas em *X* e *Y* da origem 0_c do sistema da câmera em relação ao sistema do robô:** com as projeções do passo 8 é possível calcular as coordenadas da origem 0_c no sistema do robô, que passou a ser chamado de $g0_c$, conforme segue:

$$X_{g0c} = P_{Xr'} + X_{g1} \quad (21)$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_{g0c} = P_{Yr'} + Yg_1 \quad (22)$$

Fonte: BORGES & CORDEIRO, (2012)

Lembrando que as coordenadas X_{g1} e Y_{g1} são as coordenadas lidas da memória do robô no passo 2.

Com isto, os sistemas já estão compatibilizados em virtude do programa já ter calculado as coordenadas da origem 0_c no sistema do robô.

A partir do passo 10, serão efetuadas operações necessárias para o cálculo do erro das medidas adquiridas pelo sistema. Na seção 2.3.4, cita-se que como os pontos utilizados na calibração (P_1 e P_2) já possuem coordenadas no sistema do robô, então para realizar o cálculo do erro basta comparar as coordenadas transportadas para o sistema do robô com as coordenadas no sistema robô. Para tanto, serão comparadas as coordenadas transportadas do ponto P_2 com suas coordenadas no sistema do robô (adquiridas no passo 4 da memória do robô). Para que seja possível realizar a comparação de coordenadas e identificar o erro (calculado no passo 12), primeiramente é necessário obter as projeções em *X* e *Y* rotacionadas do ponto P_2. Para isso, as mesmas operações efetuadas para o ponto P_1 nos passos 7 e 8 serão efetuadas nos passos 10 e 11 para o ponto P_2, como descrito a seguir.

10) **Cálculo da distância e azimute do alinhamento $P_0 - P_2$ no sistema da câmera:** são realizadas as mesmas operações do passo 7, porém para o ponto P_2. A Figura 61 ilustra os dados obtidos neste passo.

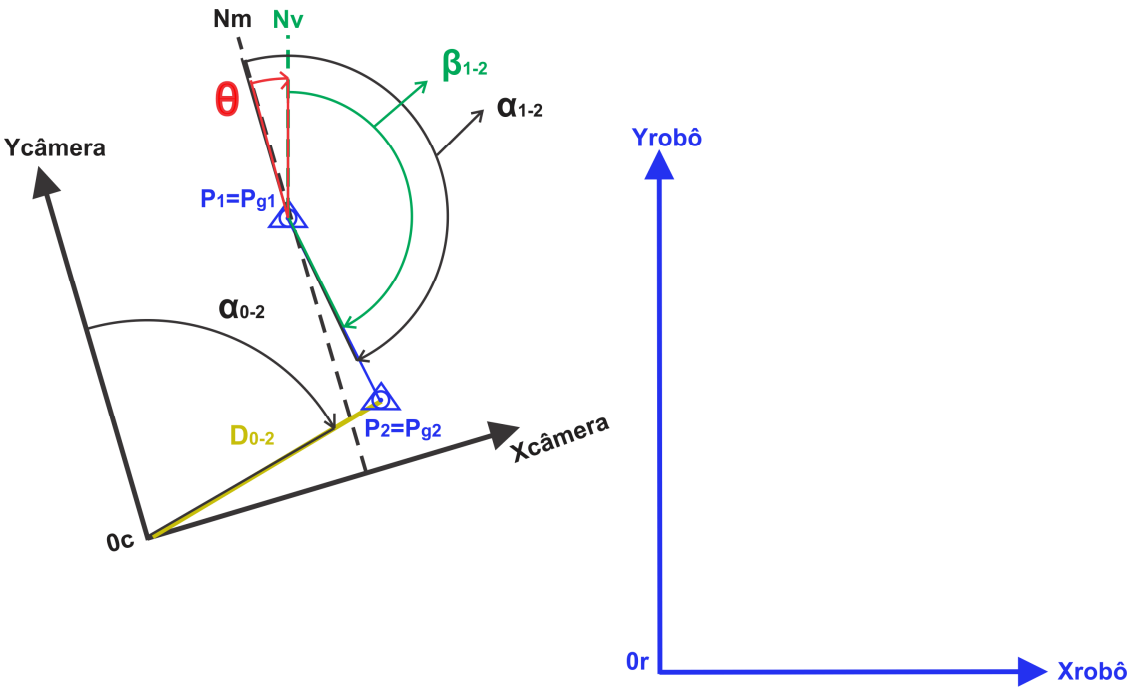


Figura 61: Azimute α_{0-2} e distância D_{0-2} do alinhamento $P_0 - P_2$ na etapa de calibração

Fonte: Autor

11) **Cálculo das projeções em X e Y rotacionadas do alinhamento $P_0 - P_2$:** são realizadas as mesmas operações do passo 8, porém para o ponto P_2. A Figura 62 ilustra os dados obtidos neste passo.

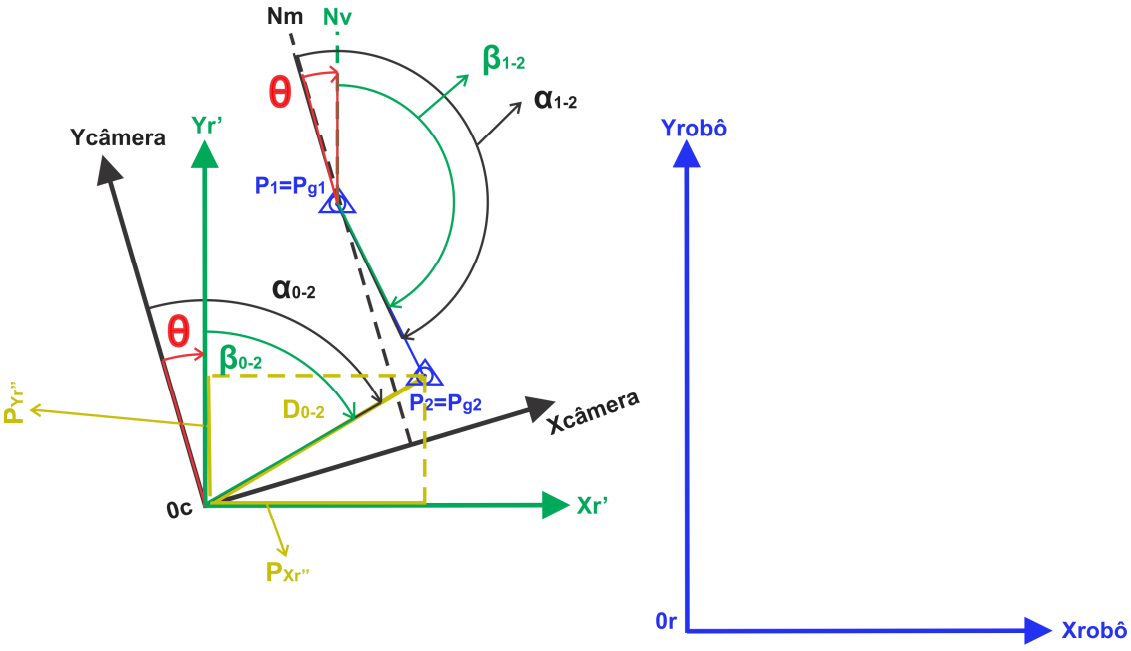


Figura 62: Projeções em X e Y rotacionadas do alinhamento $P_0 - P_2$ na etapa de calibração

Fonte: Autor

12) **Cálculo dos erros em *X* e *Y* do sistema:** com os dados obtidos até agora, já é possível realizar a comparação das coordenadas transportadas do ponto P_2 com suas coordenadas no sistema do robô (fornecidas pelo robô no passo 4 - Xg_2, Yg_2) e assim obter os erros em X e Y do sistema. Primeiramente, deve-se calcular as coordenadas transportadas (X_{g2c}, Y_{g2c}) somando as projeções rotacionadas ($P_{Xr''}, P_{Yr''}$) obtidas no passo 11 com as coordenadas da origem no sistema do robô (X_{g0c}, Y_{g0c}) obtidas no passo 9, conforme segue:

$$X_{g2c} = P_{Xr''} + X_{g0c} = X_{g2} \quad (23)$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_{g2c} = P_{Yr''} + Y_{g0c} = Y_{g2} \quad (24)$$

Fonte: BORGES & CORDEIRO, (2012)

Com isto, basta realizar a comparação das coordenadas e obter os erros em X e Y, conforme segue:

$$E_x = X_{g2} - X_{g2c} \quad (25)$$

Fonte: Autor

$$E_y = Y_{g2} - Y_{g2c} \quad (26)$$

Fonte: Autor

Os erros em *X* e *Y* serão compensados durante o processo de execução do programa de manipulação, o qual será descrito a seguir. O passo 12 finaliza a etapa de calibração e o sistema já está preparado para entrar em execução, conforme a Figura 50 mostrada na parte inicial deste capítulo.

No apêndice E são mostrados os comandos utilizados para enviar instruções ao robô RV-M1. Para a etapa de calibração foram utilizados os comandos MO (*Move*) para as posições pré-gravadas, DW (*Draw*) para realizar movimentos curtos na vertical, GC (*Grip Close*) e GO (*Grip Open*) para abrir ou fechar a garra, SP (*Speed*) para determinar a velocidade com que cada movimento será executado, o WH (*Where*) para ler as coordenadas da posição atual do robô, o PR (*Position Read*) para ler as coordenadas de uma posição específica, o ER (*Error*) para verificar se houve erros de execução e o RN (*Run*) para executar o programa.

3.5.3 Execução

A etapa de execução é responsável por enviar instruções ao robô (utilizando os dados da calibração) que o habilita manipular a peça cilíndrica em qualquer posição (coordenadas desconhecidas) dentro do campo de visão da câmera e de sua área de trabalho. O fluxo de execução é ilustrado pela Figura 63.

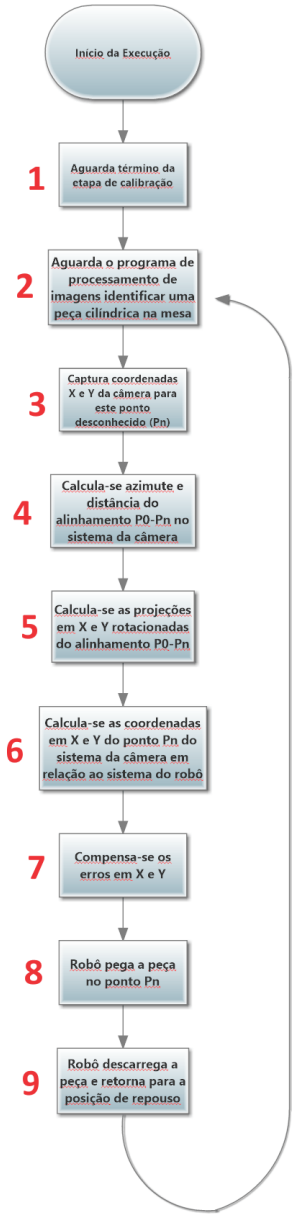


Figura 63: Fluxo de execução

Fonte: Autor

Ao observar o fluxo de execução da Figura 63, percebe-se que a etapa de calibração é dividida em 9 passos descritos a seguir:

1) **Aguarda término da etapa de calibração:** para o programa de manipulação poder efetivamente entrar em execução, primeiro deve-se esperar o término da etapa de calibração, onde são obtidos todos os dados necessários para o processo de execução do sistema. Em seguida, imediatamente o passo 2 é executado.
2) **Aguarda o programa de processamento de imagens identificar uma peça cilíndrica na mesa:** neste passo o programa de manipulação está monitorando o estado final do programa de processamento de imagens. Apenas quando o estado estiver com o valor "*TRUE*" (verdadeiro) o programa prosseguirá para o passo 3. Quando o estado está em "*TRUE*" significa que existe uma peça cilíndrica válida sobre a mesa, caso contrário significa ou que não existe peça alguma ou algum objeto de características inválidas (diâmetro, formato, comprimento, dentre outras) está sobre a mesa. Neste passo, o operador do sistema coloca a peça em um lugar qualquer do plano da mesa (dentro do campo de visão da câmera e da área de trabalho do robô), como ilustra a Figura 64.
3) **Captura as coordenadas em *X* e *Y* da câmera para este ponto desconhecido (P_n):** como a peça reconhecida no passo 2 pode estar localizada em qualquer lugar do campo de visão da câmera (Figura 64), então pode-se considerar que suas coordenadas são desconhecidas para o sistema de coordenadas do robô, o que origina um ponto desconhecido no espaço bidimensional chamado P_n, como ilustra a Figura 65.

De forma similar ao passo 2 da etapa de calibração descrita anteriormente, o programa de manipulação, sendo executado no *LabVIEW*, captura as coordenadas lidas pelo programa de processamento de imagens, sendo executado no *Vision Builder*, adquirindo assim as coordenadas não georreferenciadas do sistema da câmera (X_n, Y_n) para o ponto Pn.

A partir do passo 4 são realizados os cálculos necessários para transportar as coordenadas do ponto P_n do sistema da câmera para o sistema do robô. Em seguida, os erros em *X* e *Y* serão compensados no passo 7.

Figura 64: Peça localizada em um ponto desconhecido

Fonte: Autor

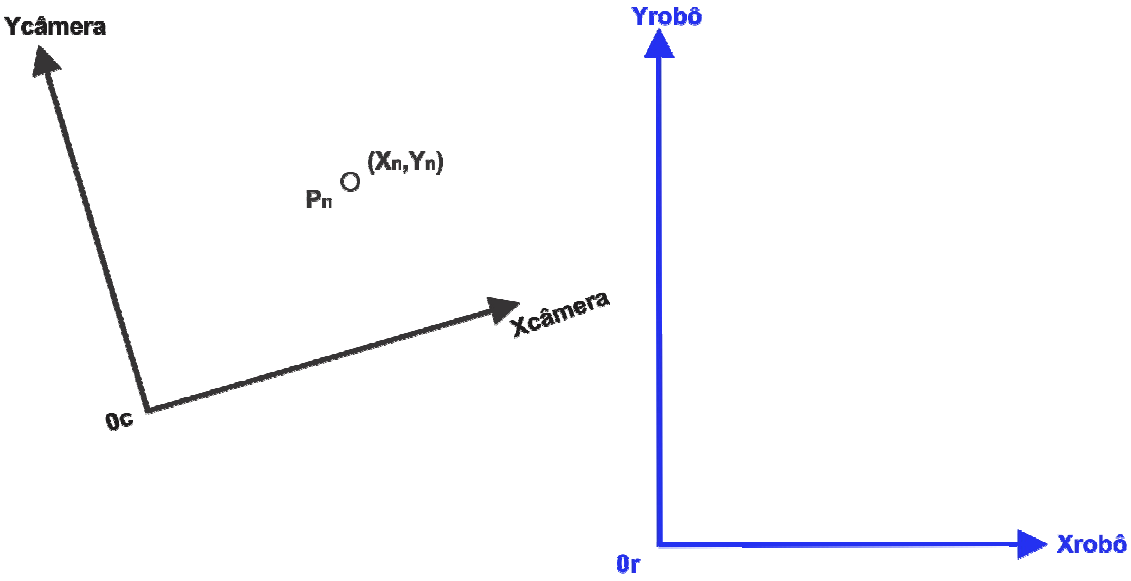


Figura 65: Ponto desconhecido P_n de coordenadas não georreferenciadas (X_n, Y_n) na etapa de execução

Fonte: Autor

4) **Cálculo do azimute e distância do alinhamento $P_0 - P_n$ no sistema da câmera:** são realizados os mesmos cálculos apresentados na seção 2.4. O objetivo é calcular as coordenadas verdadeiras do ponto P_n. Para tanto,

primeiramente deve-se calcular o azimute α_{0-n} do alinhamento $P_0 - P_n$ de forma similar ao passo 5 da etapa de calibração.

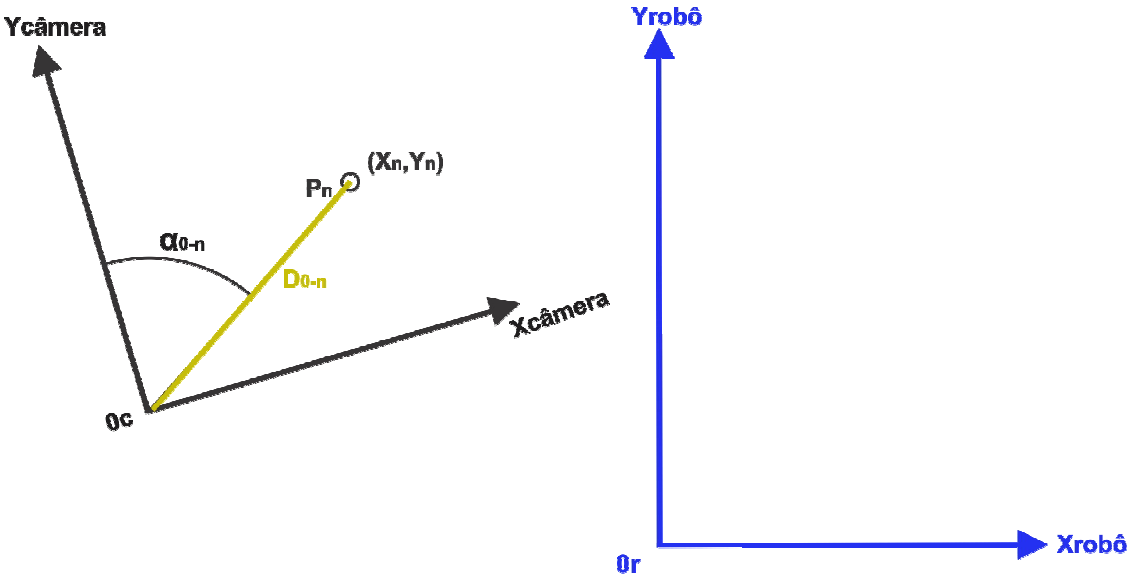


Figura 66: Azimute e distância do alinhamento na etapa de execução

Fonte: Autor

Feito isto, é necessário calcular a distância do alinhamento $P_0 - P_n$ (ilustrado na Figura 66, destacado em amarelo) conforme abaixo:

$$D_{O-n} = \sqrt{(X_n - X_0)^2 + (Y_n - Y_0)^2} \quad (27)$$

Fonte: BORGES & CORDEIRO, (2012)

5) **Cálculo das projeções em *X* e *Y* rotacionadas do alinhamento $P_0 - P_n$:** continuando os cálculos da seção 2.12.3, primeiramente é necessário realizar o cálculo do azimute no sistema do robô do alinhamento $P_0 - P_n$ para que as projeções rotacionadas $P_{Xr'}$ e $P_{Yr'}$ possam ser calculadas. A sequência de cálculos é descrita a seguir.

Com o α_{0-n} calculado no passo 7, pode-se calcular o β_{0-n}(ilustrado na Figura 67) da seguinte forma (onde θ é a rotação calculada anteriormente na etapa de calibração no passo 6):

$$\beta_{0-n} = \alpha_{0-n} - \theta \quad (28)$$

Fonte: BORGES & CORDEIRO, (2012)

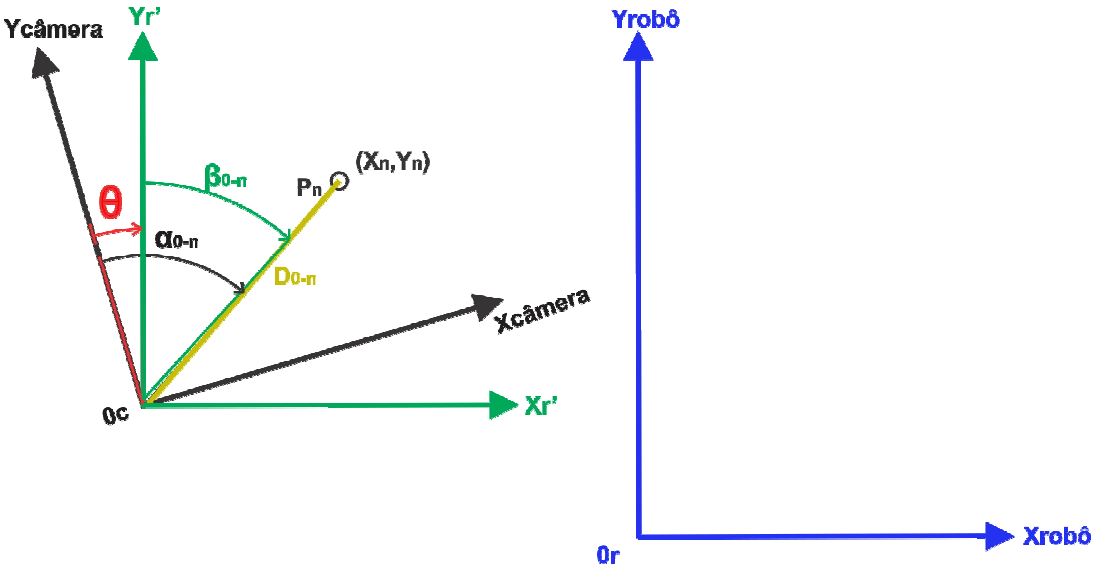


Figura 67: Azimute "georreferenciado" do alinhamento $P_0 - P_n$ na etapa de execução

Fonte: Autor

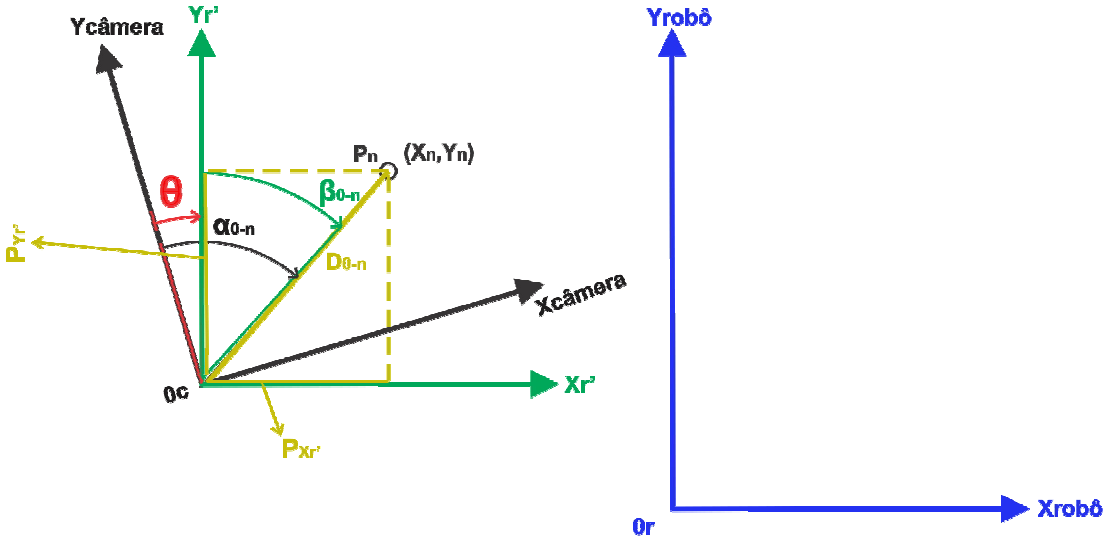


Figura 68: Projeções em X e Y rotacionadas do alinhamento $P_0 - P_n$ na etapa de execução

Fonte: Autor

Observa-se na Figura 67, um sistema de coordenadas $(X_{r'},Y_{r'})$ destacado na cor verde que é o resultado da rotação do sistema da câmera para o do robô. Com este azimute β_{0-n} e a distância D_{O-n} (calculada no passo 4) calcula-se as projeções em X e em Y ($P_{Xr'}$ e $P_{Yr'}$, ilustradas na Figura 68) do alinhamento $P_0 - P_n$ no sistema verdadeiro, da seguinte forma:

$$P_{Xr'} = D_{O-n} \operatorname{sen}(\beta_{0-n}) \quad (29)$$

Fonte: BORGES & CORDEIRO, (2012)

$$P_{Yr'} = D_{O-n} \cos(\beta_{0-n}) \quad (30)$$

Fonte: BORGES & CORDEIRO, (2012)

6) **Cálculo das coordenadas em X e Y do ponto P_n do sistema da câmera em relação ao sistema do robô:** com as projeções rotacionadas ($P_{Xr'}$,$P_{Yr'}$) do passo 5 é possível calcular as coordenadas do ponto P_n no sistema do robô (X_{gnc},Y_{gnc}), que passou a ser chamado de P_{gn}, somando suas projeções com as coordenadas da origem P_0 (X_{g0c},Y_{g0c}) do sistema da câmera em relação ao sistema do robô (calculadas no passo 9 da etapa de calibração), conforme segue:

$$X_{gnc} = P_{Xr'} + X_{g0} \quad (31)$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_{gnc} = P_{Yr'} + Y_{g0} \quad (32)$$

Fonte: BORGES & CORDEIRO, (2012)

7) **Compensa-se os erros em X e Y:** se as coordenadas X_{gnc} e Y_{gnc} obtidas no passo 6 fossem utilizadas pelo programa de manipulação para enviar instruções para o robô manipular a peça no ponto P_n no passo 8, os erros em *X* e *Y* (E_x,E_y) calculados no passo 12 da etapa de calibração ainda não teriam sido compensados, com isso, é muito provável que o robô erre o seu posicionamento ao tentar manipular a peça. Portanto, é necessário compensar tais erros corrigindo as coordenadas (X_{gnc},Y_{gnc}) conforme as equações a seguir.

$$X_{gncCorrigida} = X_{gnc} + E_x \quad (33)$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_{gncCorrigida} = Y_{gnc} + E_y \quad (34)$$

Fonte: BORGES & CORDEIRO, (2012)

8) **Robô pega a peça no ponto P_n:** agora que os erros em *X* e *Y* já foram compensados no passo 7, então o programa de manipulação já pode enviar as coordenadas ($X_{gncCorrigida}$,$Y_{gncCorrigida}$) ao robô para que o mesmo

manipule a peça localizada no ponto P_n. Com isso, o robô sai de sua posição de repouso e pega a peça no ponto desconhecido P_n, como ilustra a Figura 69.

Figura 69: Robô pega a peça no ponto desconhecido

Fonte: Autor

9) **Robô descarrega a peça e retorna para sua posição de repouso:** após o robô pegar a peça localizada no ponto P_n, através de instruções enviadas no passo 8, o mesmo descarrega a peça em uma posição pré-gravada em sua memória pelo operador do sistema e retorna para sua posição de repouso.

Com o término do último passo 9, o sistema já está pronto para receber uma nova peça em um ponto desconhecido dentro do campo de visão da câmera e da área de trabalho do robô, por este motivo, como ilustra o fluxo de execução da Figura 63, o programa de manipulação retorna para o passo 2 que aguarda o programa de processamento de imagens identificar uma nova peça cilíndrica sobre a mesa, recomeçando todo o processo de execução novamente.

Os comandos utilizados para enviar instruções ao robô RV-M1 durante a etapa de execução são os mesmos utilizados na etapa de calibração.

No apêndice F são apresentados conceitos em robótica para realizar os mesmos cálculos apresentados nas etapas de calibração e execução. Isto é, realizar a rotação e translação entre os sistemas da câmera e do robô.

3.5.4 O painel frontal e o diagrama de blocos

Um programa no *LabVIEW* é dividido em duas seções: o painel frontal e o diagrama de blocos. No painel frontal do *LabVIEW* são mostradas os valores e estados de todas as variáveis que estão sendo utilizadas dentro do programa. No programa de manipulação o painel frontal foi dividido em quatro seções nomeadas como comunicação, calibração, sistema de visão e manipulação da peça. As Figuras

70, 72, 73 e 74 mostram todas as seções que dividem o painel frontal. A seção de comunicação possibilita ao usuário configurar em qual porta serial do computador se deseja estabelecer comunicação com a central do robô RV-M1. A seção de calibração permite a leitura do usuário dos valores de todas as variáveis e instruções pertencentes à etapa de calibração. A seção de sistema de visão contém em tempo real todas as informações adquiridas pelo programa de processamento de imagens, assim, todas as variáveis fornecidas através do *Vision Builder* são lidas, o que possibilita ao usuário observar o resultado final de cada inspeção realizada. Por fim, a seção de manipulação da peça contém os valores de todas as variáveis e instruções pertencentes a esta fase de execução, onde o robô manipula a peça em um ponto desconhecido.

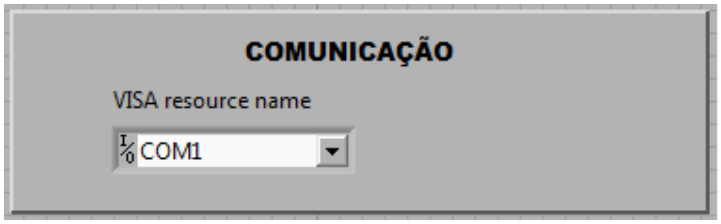


Figura 70: Seção de comunicação do programa de manipulação

Fonte: Autor

Foi encontrado um programa exemplo no *LabVIEW* para comunicação do computador com sua porta serial RS-232. Este programa foi adaptado com as configurações requeridas pelo robô, conforme ilustra a Figura 71, tornando possível a comunicação entre o computador e o robô através do programa do *LabVIEW*.

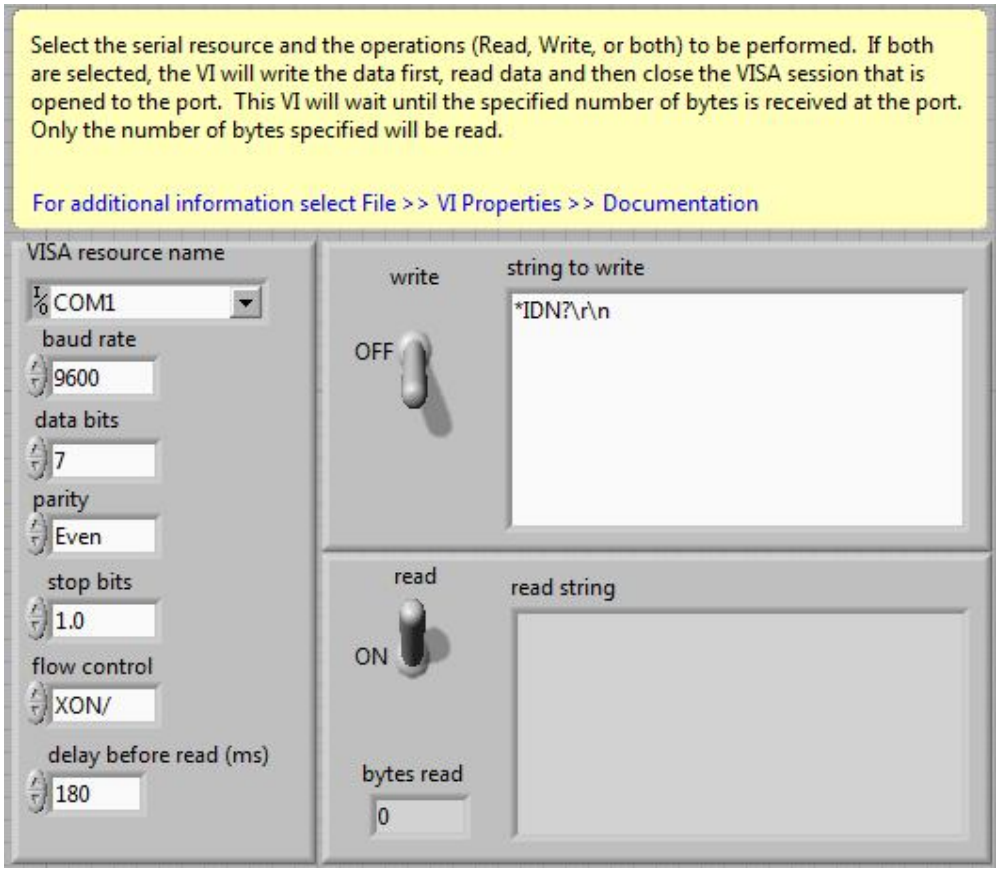


Figura 71: Programa exemplo do *LabVIEW* adaptado para comunicar o computador com o robô

Fonte: Autor

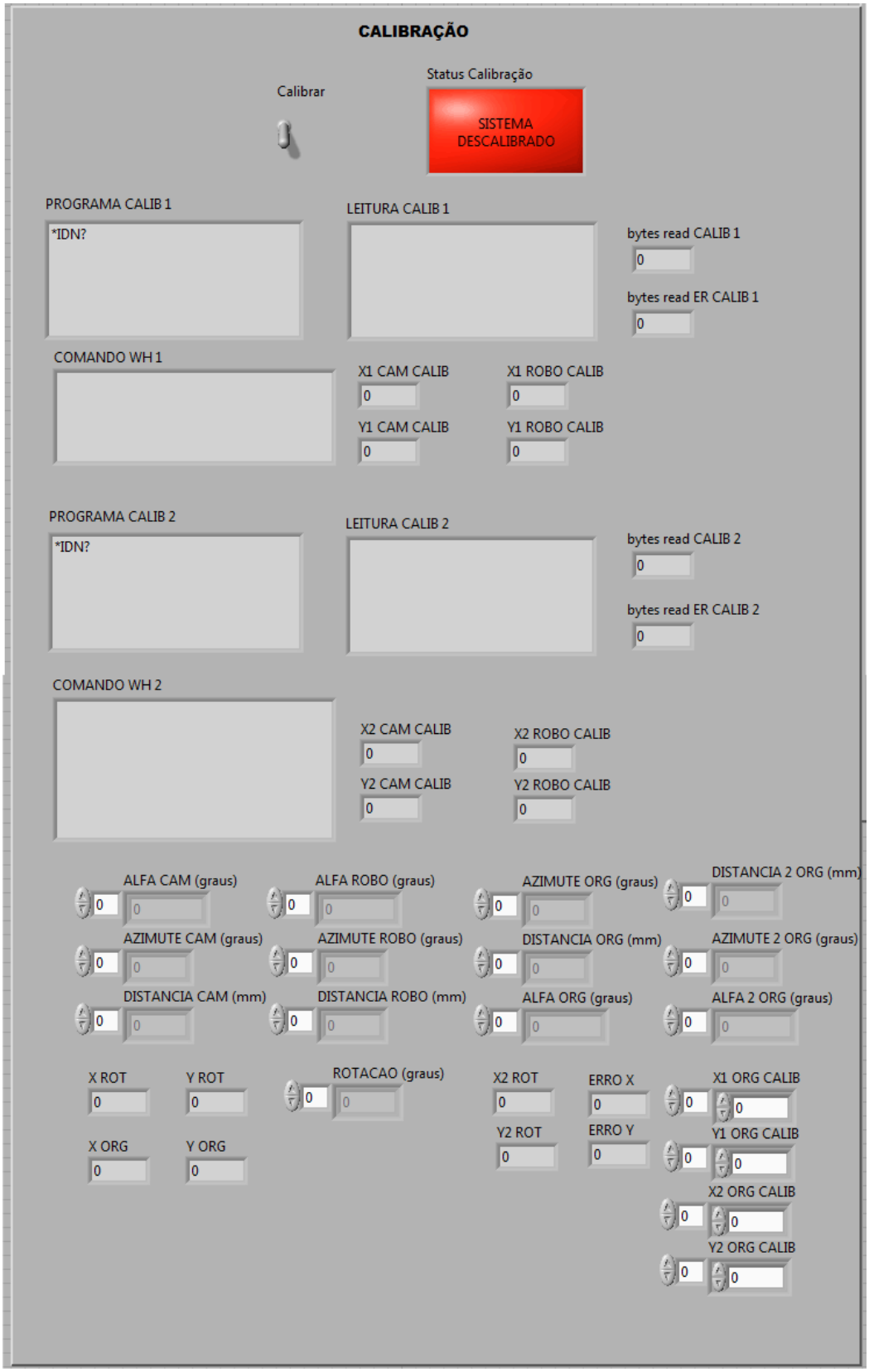


Figura 72: Seção de calibração do programa de manipulação

Fonte: Autor

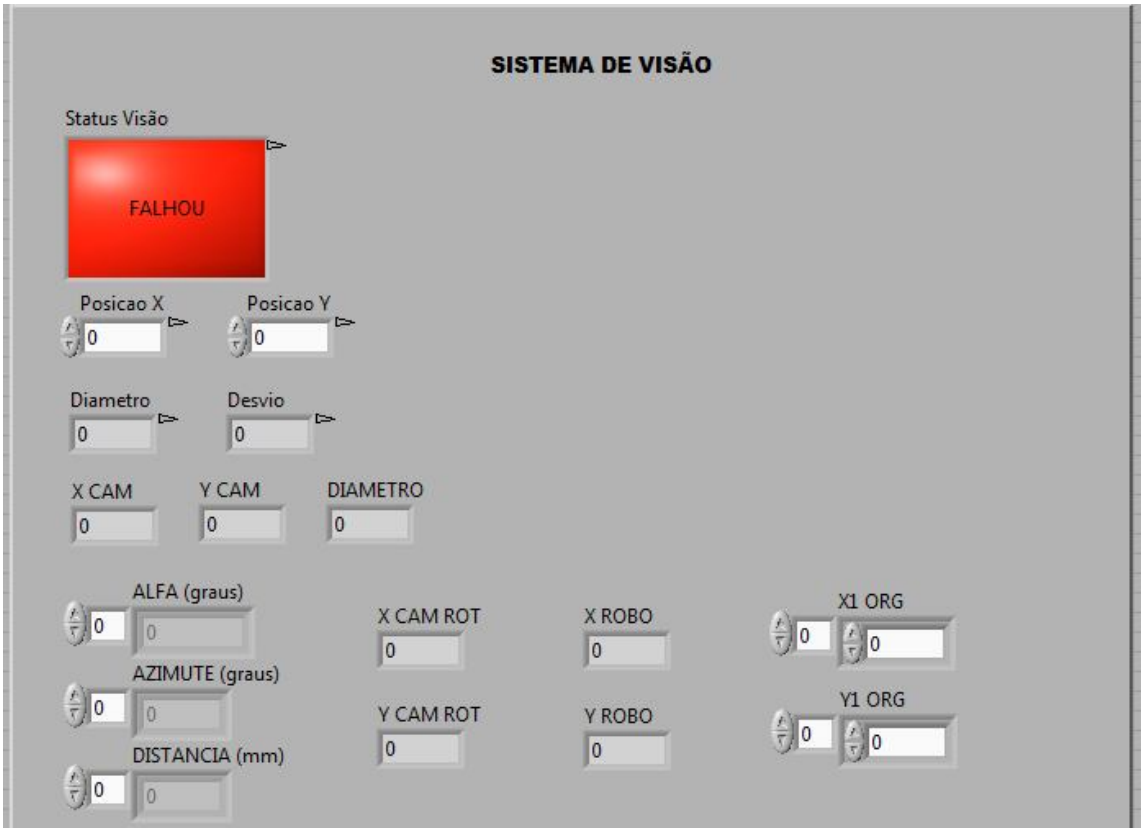


Figura 73: Seção do sistema de visão do programa de manipulação

Fonte: Autor

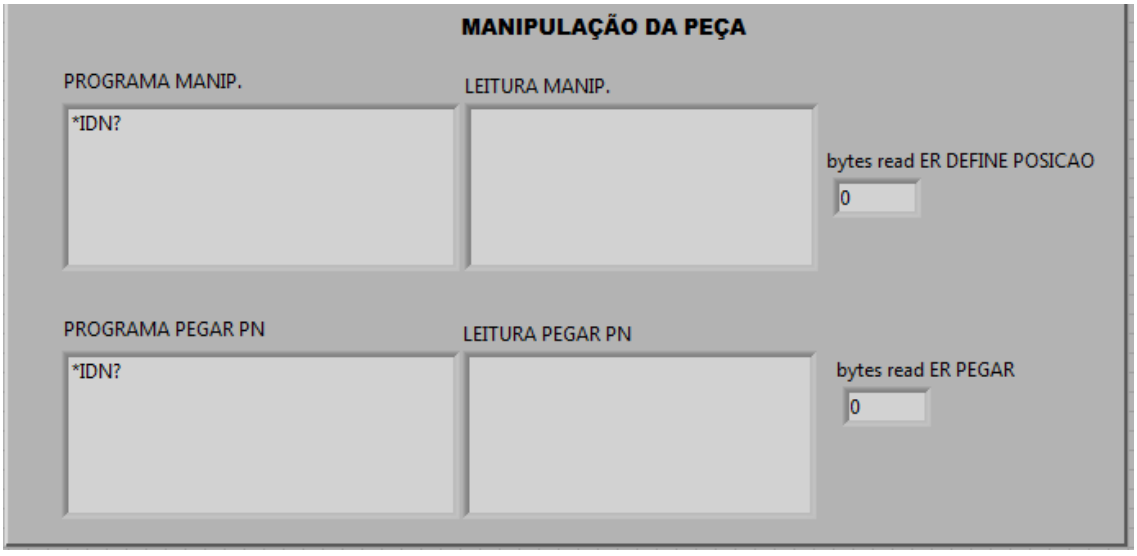


Figura 74: Seção de manipulação da peça do programa de manipulação

Fonte: Autor

Conforme descrito na seção 3.2.5, no *LabVIEW* o programa é estruturado através de vários blocos, onde cada um realiza uma determinada tarefa. Tais blocos

são apresentados na seção de diagrama de blocos da interface do *LabVIEW*. Como exemplo, a Figura 75 apresenta o diagrama de blocos do começo da etapa de execução do programa definida pelas etapas de 1 a 3 descritas anteriormente.

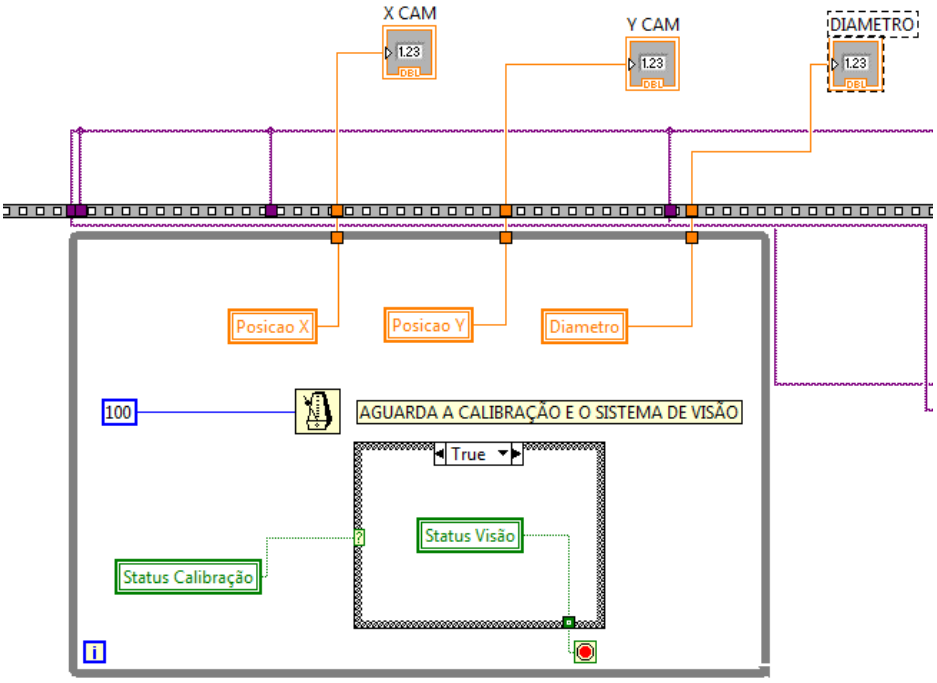


Figura 75: Exemplo de um diagrama de blocos no *LabVIEW*

Fonte: Autor

4. CONSIDERAÇÕES FINAIS

Este projeto possibilitou adquirir e aprimorar conhecimentos sobre robótica, mecânica, comunicação serial, processamento de imagens, visão artificial, programação em ambiente *LabVIEW* e *Vision Builder* e especialmente topografia e geodésia, o qual foi um assunto totalmente novo para o autor deste trabalho.

Em geral, o projeto cumpriu o seu objetivo, tendo em vista que o sistema de visão robótico possibilita a manipulação de peças cilíndricas pelo robô em pontos desconhecidos dentro de sua área de trabalho, permite que ações sejam tomadas apenas após a identificação de eventos externos (identificação da peça, retirada da peça) em torno do espaço de trabalho e aumenta a flexibilidade de funcionamento do robô, em virtude de não exigir ao mesmo estar paralelo ou alinhado em relação ao plano em que o objeto a ser manipulado se localiza.

Foram constatadas algumas irregularidades funcionais no robô, tais como:

- Vibrações durante a movimentação do braço mecânico: quando o robô se movimenta, vibrações interferem durante todo o processo de movimentação das juntas.

- **Limitações nas juntas e freios:** devido às políticas da empresa Mitsubishi, a manutenção do robô pode ser apenas realizada por técnicos autorizados pela companhia, os quais são dificilmente encontrados pela universidade. Consequentemente, torna-se impossível manter o estado ideal das partes mecânicas do manipulador.

- **Problemas de repetibilidade e precisão:** com o decorrer do tempo as tarefas de manipulação das peças não são executadas da mesma maneira que no início do processo, ocasionando imprecisões e erros de movimentação que podem comprometer toda a confiabilidade das operações requisitadas ao robô.

Levando em conta tais fatores, concluiu-se que é apropriado implantar um novo elemento sensorial para o robô, um sistema de visão robótico.

No início do projeto, encontraram-se dificuldades em encontrar uma solução para viabilizar a manipulação da peça em um ponto desconhecido. Durante a fase de pesquisa conhecimentos em topografia e geodésia foram adquiridos, os quais foram satisfatórios para solucionar tal problema, além de viabilizar uma das flexibilidades oferecidas por este trabalho (descrita anteriormente) através da

compatibilização de sistemas de coordenadas diferentes. Também, comparado com a proposta inicial deste projeto, a programação não foi feita integralmente no *LabVIEW* conforme proposto, tendo em vista a facilidade que o *Vision Builder* ofereceu em criar um programa que processe e inspecione imagens com alto desempenho computacional.

5. RECOMENDAÇÕES PARA TRABALHOS FUTUROS

São sugeridas as seguintes melhorias para este projeto:

- Manipular peças de formatos diferentes;
- Manipular peças de alturas diferentes, exigindo que o sistema funcione em três dimensões (3D);
- Implementar ações para a segurança do sistema, identificando eventos externos que possam interferir no processo de execução, tais como obstáculos que interfiram na movimentação do robô;
- Adaptar o sistema para diferentes tipos de iluminação no ambiente;
- Possibilitar ao sistema de visão analisar e manipular várias peças ao mesmo tempo.

REFERÊNCIAS

BORGES, Adilson L.; CORDEIRO, Arildo D. **Apostila de Topografia na Prática;** disponível em http://pessoal.utfpr.edu.br/arildo/?id=6, acessado em 02/11/2012.

BOVIK, Alan C. (Ed.). **The essential guide to image processing.** 2nd ed. London; Boston: Academic Press/Elsevier, 2009. xxi, 853 p.

CAMBRIDGE IN COLOUR. **"*Histogramas – Parte 1*";** disponível em http://www.cambridgeincolour.com/pt-br/tutorials/pt-br-histograms1.htm, acessado em 10/06/2011.

CRAIG, John J. **Robótica.** 3ª ed. São Paulo: Pearson Education do Brasil, 2012, 379 p.

FILHO, Roberto A. **Padrão serial RS-232**; disponível em http://www2.eletronica.org/artigos/eletronica-digital/padrao-serial-rs-232, acessado em 08/03/2013.

FORESTI, Renan L. **Sistema de visão robótica para reconhecimento de contornos de componentes na aplicação de processos industriais.** Porto Alegre: UFRGS/RS, 2006. 64 f. Dissertação (Mestrado) – Universidade Federal do Rio Grande do Sul, Curso de Pós-Graduação em Engenharia Mecânica.

GONZALEZ, Rafael C.; WOODS, Richard E. **Processamento de imagens digitais.** São Paulo: E. Blücher, 2000. 509 p.

GONZALEZ, Rafael C.; WOODS, Richard E. **Digital image processing.** 3ª ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2008. 954 p.

GRASSI, Maurício V. **Desenvolvimento e aplicação de um sistema de visão para robô industrial de manipulação.** Porto Alegre: UFRGS/RS, 2005. 85 f. Dissertação (Mestrado) – Universidade Federal do Rio Grande do Sul, Curso de Pós-Graduação em Engenharia Mecânica.

GROOVER, Mikell P. **Automation, production systems, and computer integrated manufacturing.** 3rd ed. UpperSaddle River: Prentice-Hall, 2008. 831 p.

JAIN A. K; PANKANTI S. L. Hong; BOLLE R. **On-line identity-authentication system using fingerprints**. *Proc. IEEE (Special Issue on Automated Biometrics)*, 1997.

JANECZKO, Cézar. **Limiarização.** Curitiba: UTFPR/PR, 2010. Disponível em http://www.pessoal.utfpr.edu.br/janeczko/index_files/pdi/aula06_PDI_Limiarizacao.pdf, acessado em 22/09/2011.

JARDIM, Laercio A. **Sistema de visão robótica para reconhecimento e localização de objetos sob manipulação por robôs industriais em células de**

manufatura. Brasília: UNB/DF, 2006. 110 f. Dissertação (Mestrado) – Universidade de Brasília, Curso de Pós-Graduação em Engenharia Mecânica.

LATECKI, Longin J. **Image Segmentation Using Region Growing and Shrinking.** Estados Unidos, 2005. Disponível em http://www.cis.temple.edu/~latecki/Courses/CIS750-03/Lectures/Siddu_RegionGrowing.ppt, acessado em 22/09/2011.

MALAQUIAS, D. F.; ASSOLARI, K.; NOGAWA, L. M. **Desenvolvimento de um sistema de controle de um braço robótico de cinco eixos.** Curitiba: UTFPR/PR, 2012. 86 f. Trabalho de conclusão de curso – Universidade Tecnológica Federal do Paraná, Curso de Tecnologia em Mecatrônica Industrial.

MARQUES FILHO, Ogê; VIEIRA NETO, Hugo. **Processamento digital de imagens**. Rio de Janeiro: Brasport, 1999. 406 p.

MCCORMAC, Jack C. (Ed.). **Topografia.** 5ª ed. Rio de Janeiro: LTC, 2007, p. 136 de 391 p. + 1 CD-ROM.

MITSUBISH. **Industrial micro-robot system : model RV-M1.** Japão, 1994. 260p.

MITSUBISHI ELECTRIC RESEARCH LABORATORIES. **Iris Recognition from 1-2 Meters.** Estados Unidos: 2007. Disponível em http://www.merl.com/areas/irisrecognition/, acessado em 22/09/2011.

NATIONAL INTRUMENTS. **NI Vision Concepts Manual.** Estados Unidos, 2005. 399p. Disponível em http://www.ni.com/pdf/manuals/372916e.pdf, acessado em 08/03/2013.

NATIONAL INTRUMENTS. **NI Vision Builder for Automated Inspection.** Estados Unidos, 2013. Disponível em http://sine.ni.com/nips/cds/view/p/lang/pt/nid/11700, acessado em 08/03/2013.

NATIONAL INTRUMENTS. **Using the LabVIEW Shared Variable.** Estados Unidos, 2013. Disponível em http://www.ni.com/white-paper/4679/en, acessado em 08/03/2013.

NICOLOSI, Denys Emílio Campion. **Microcontrolador 8051 detalhado**. 3. ed. São Paulo: Érica, 2002.

PAVIM, Alberto X. **Contribuições na otimização de um sistema de visão para detecção, medição e classificação automática do desgaste de ferramentas de corte em processos de usinagem.** Florianópolis: UFSC/SC, 2005. 236 f. Dissertação (Mestrado) – Universidade Federal do Santa Catarina, Curso de Pós-Graduação em Engenharia Elétrica.

PPAINTINGA. **"*Grayscale watercolor paintings*";** disponível em http://ppaintinga.com/grayscale-watercolor-paintings/, acessado em 07/10/2012.

ROSÁRIO, João Maurício. **Princípios de mecatrônica.** São Paulo: Prentice-Hall, 2005. 356 p.

SISTEMAS INDUSTRIAIS INTELIGENTES – S2I. **Apostila de Sistemas de Visão.** Florianópolis, 2005. Disponível em http://s2i.das.ufsc.br/harpia/downloads/apostila-sistemas-visao.pdf, acessado em 13/02/2013.

SOARES, Marcio J. Comunicação RS-232 - Noções básicas - Parte 1. **Revista Saber Eletrônica**, São Paulo, mai. 2008. Disponível em http://www.sabereletronica.com.br/secoes/leitura/774, acessado em 08/03/2013.

SOLOMON, Chris; BRECKON, Toby. **Fundamentals of Digital Image Processing: A Practical Approach with Examples in Matlab.** Reino Unido: Wiley Blackwell, 2011. 355 p.

APÊNDICE A – Sistemas de classificação

Solomon (2011) explica que elaborar sistemas automáticos de classificação é um meio do ser humano conseguir encontrar uma solução para um determinado problema de processamento de imagens com praticidade e boa relação custo-benefício. Essencialmente, duas áreas se enquadram na elaboração destes sistemas: a especificação da tarefa e a rotulação de classes. A especificação da tarefa deve responder a seguinte pergunta: O que exatamente o classificador deve reconhecer? Para tanto, é necessário decidir quais classes devem ser consideradas e quais variáveis ou parâmetros são importantes para encontrar a classificação adequada. No exemplo citado anteriormente, para um diagnóstico médico preliminar, classificar as células da lâmina histológica entre 'normais' ou 'anormais' é suficiente. Porém, dependendo das informações de formato, densidade, tamanho e cor das células na imagem, este mesmo classificador pode as células 'anormais' de maneira mais acurada, atribuindo a mais categorias, tais como 'anêmicas' e 'tipo A de infecção'. Já a rotulação de classes consiste no processo de atribuir manualmente rótulos no estágio inicial do reconhecimento de objetos. Sendo assim, são atribuídos exemplos reais com classes específicas baseado em propriedades notáveis na imagem.

Segundo Solomon (2011), um simples exemplo de reconhecimento e interpretação é classificar objetos que saem de uma máquina processadora de alimentos entre três tipos: pinhole, lentilhas e sementes de abóbora. Um exemplo da imagem a ser analisada é ilustrado na Figura A.1.

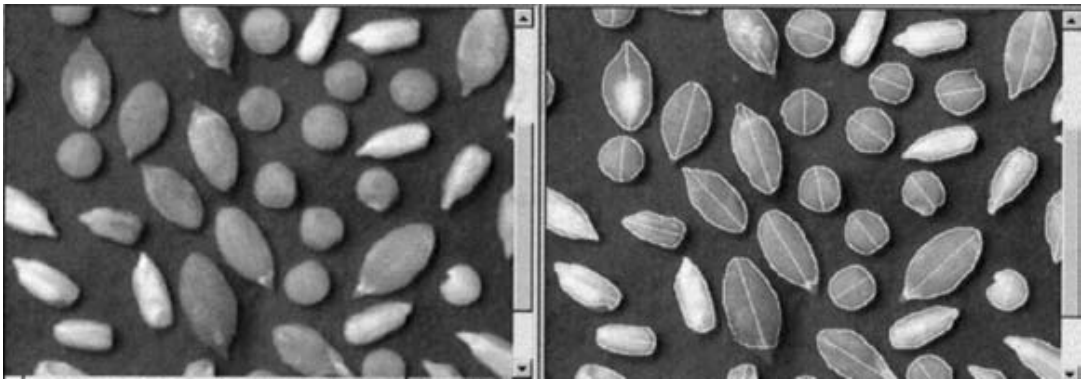


Figura A.1: Os três tipos de objetos: pinhole, lentilhas e sementes de abóbora. A imagem da direita foi processada para obter as características de circularidade e erro da linha de ajuste

Fonte: SOLOMON, (2011)

É considerado que o processamento executado nas imagens da Figura A.1 pôde obter de maneira confiável, medidas dos objetos quanto a duas características: erro da linha de ajuste e circularidade. Esta etapa é chamada de extração de características. Para qualquer um dos três objetos a serem analisados é obtido um vetor bidimensional de características o qual se pode indicar, neste caso específico, como um ponto em um espaço bidimensional de características, como ilustra a Figura A.2, onde pinhole, lentilhas e sementes de abóbora são respectivamente identificadas como losangos, quadrados e triângulos.

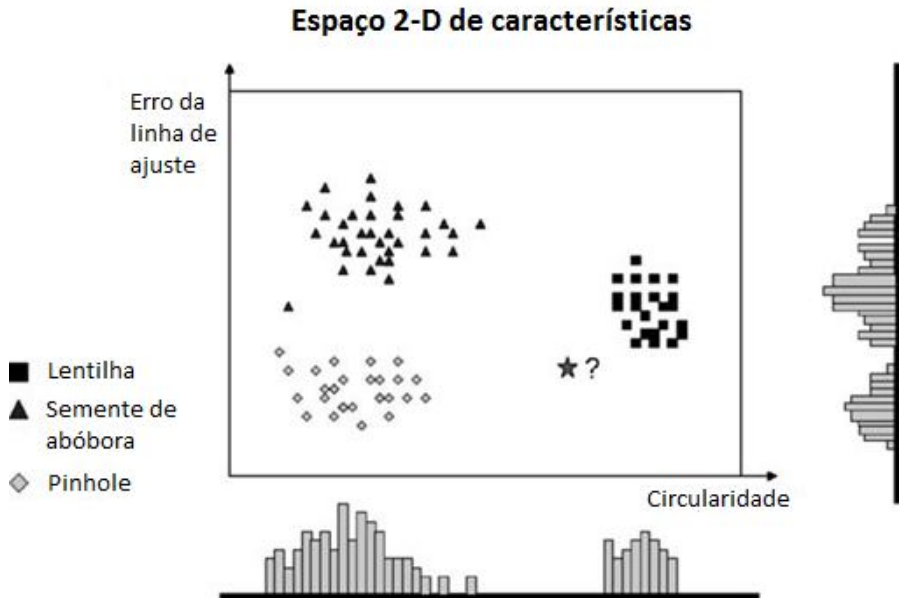


Figura A.2: Espaço de características bidimensional que discrimina os objetos. A representação de única dimensão mostra gráficos de barra respectivos às características de seus respectivos eixos *x* e *y*

Fonte: Traduzido de SOLOMON, (2011)

Ao analisar a Figura A.2, se percebe que as três classes formam diferentes aglomerados no espaço de características, o que indica que as duas características escolhidas, erro da linha de ajuste e circularidade, são adequadas para discriminar estas diferentes classes. Porém, se considerar apenas uma das características escolhidas, como ilustram os gráficos de barra, uma sobreposição de classes acontece na análise de algumas porções dos objetos da imagem. Mas, mesmo utilizando duas características, há um objeto no espaço que tem um vetor de características representado por uma estrela. Para atribuí-lo a alguma classe, a

maneira mais sensata é defini-lo como uma lentilha, devido à proximidade com esta classe. Porém, conforme Solomon (2011), para ter certeza dessa definição seria necessário utilizar outros métodos de classificação, tais como o da mínima distância e o bayesiano, os quais utilizam recursos matemáticos avançados, como funções lineares discriminantes e o algoritmo *k-means*. Tais métodos não serão abordados neste trabalho.

Com isso, neste exemplo, o reconhecimento cumpre a tarefa de rotular cada objeto ou como amêndoa de pinho ou lentilha ou semente de abóbora. Já a interpretação, cumpre a tarefa de atribuir um significado ao conjunto de objetos, como por exemplo, de que existem seis sementes de abóbora, oito lentilhas, cinco amêndoas de pinho e um objeto que se aproxima de ser uma lentilha. Ao aplicar tais conceitos no presente trabalho, as mesmas características usadas no exemplo anterior podem ser utilizadas (circularidade e erro da linha de ajuste), o que resulta em o reconhecimento rotular o objeto da imagem como uma peça cilíndrica e a interpretação estabelecer que existe apenas uma peça cilíndrica e mais nenhum outro objeto na imagem.

APÊNDICE B – Transporte de coordenadas

Para transformar as coordenadas arbitradas em levantamentos topográficos de terrenos com orientação magnética em coordenadas verdadeiras, pode-se realizar o georreferenciamento dos pontos levantados ou, quando não se dispõe dos equipamentos necessários, podem-se transportar as coordenadas de pontos geodésicos existentes nas proximidades. Esta última operação será demonstrada a seguir.

Na Figura B.1, o alinhamento formado pelos pontos P_1 e P_2 representa um dos lados de uma poligonal qualquer de um terreno levantado topograficamente com orientação magnética (bússola), o que resulta em seu sistema de coordenadas ser baseado na linha norte/sul magnética (destacado em preto). Os pontos P_{G1} e P_{G2} são geodésicos (pertencentes ao Sistema Geográfico Brasileiro - SGB), o que resulta em seu sistema de coordenadas ser baseado na linha norte/sul verdadeira (destacado em azul). As coordenadas dos pontos P_{G1} e P_{G2} podem ser transportadas para os pontos P_1 e P_2 medindo-se o ângulo θ_1 formado pelo alinhamento $P_{G1} - P_{G2}$ e o alinhamento $P_{G1} - P_1$. Considerando a distância d_1 entre os pontos P_{G1} e P_1, é possível calcular o azimute do alinhamento $P_{G1} - P_1$ α_{G1-1}. Primeiramente, obtém-se o azimute do alinhamento $P_{G2} - P_{G1}$ (β_{2-1}), que é o contra azimute do alinhamento $P_{G1} - P_{G2}$ (β_{1-2}), (equação B.1). A partir disto, aplicam-se os três passos para cálculo de azimute descrito anteriormente (equação B.2).

$$\beta_{2-1} = \beta_{1-2} + 180° \tag{B.1}$$

Fonte: BORGES & CORDEIRO, (2012)

$$\alpha_{G1-1} = \begin{cases} se\ \beta_{2-1} + \theta_1 > 360 = (\beta_{2-1} + \theta_1) - 360 \\ se\ \beta_{2-1} + \theta_1 > 180 = (\beta_{2-1} + \theta_1) - 180 \\ se\ \beta_{2-1} + \theta_1 < 180 = (\beta_{2-1} + \theta_1) + 180 \end{cases} \tag{B.2}$$

Fonte: BORGES & CORDEIRO, (2012)

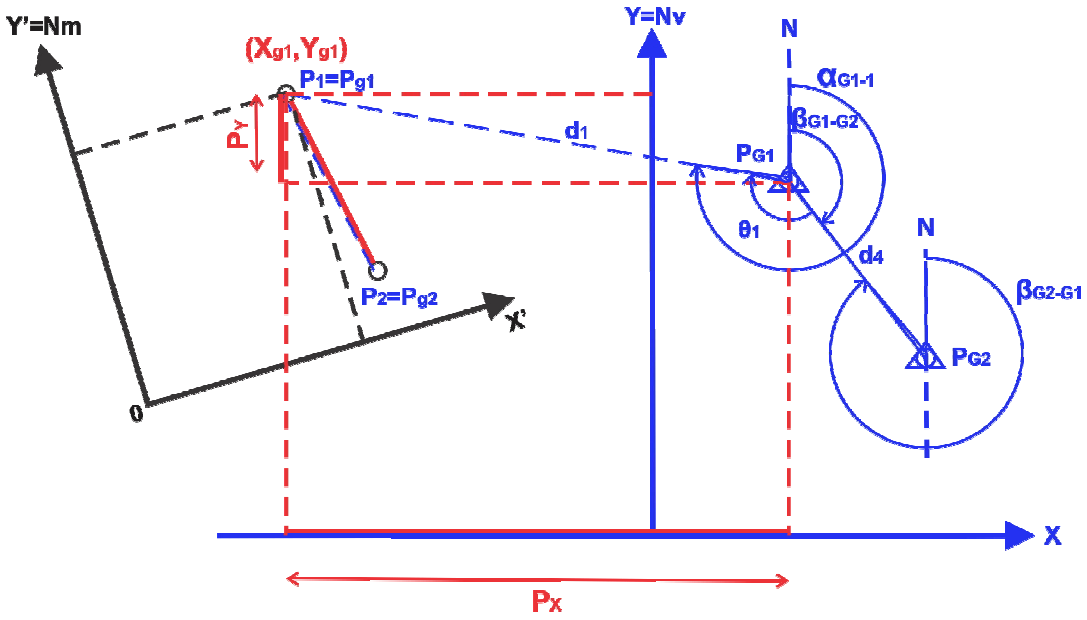


Figura B.1: Transporte das coordenadas do ponto P1
Fonte: Autor

Com este azimute α_{G1-1} e a distância d_1 calcula-se as projeções em *X* e em *Y* do alinhamento $P_{G1} - P_1$ no sistema verdadeiro, da seguinte forma:

$$P_x = d_1 \operatorname{sen}(\alpha_{G1-1}) \tag{B.3}$$

Fonte: BORGES & CORDEIRO, (2012)

$$P_y = d_1 \cos(\alpha_{G1-1}) \tag{B.4}$$

Fonte: BORGES & CORDEIRO, (2012)

Com essas projeções e as coordenadas georreferenciadas (X_{G1},Y_{G1}) do ponto P_{G1} calcula-se as coordenadas (X_1,Y_1) verdadeiras do ponto P_1 que passa a ser chamado P_{g1} por estar no sistema verdadeiro da seguinte forma:

$$X_1 = P_x + X_{G1} = X_{g1} \tag{B.5}$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_1 = P_y + Y_{G1} = Y_{g1} \tag{B.6}$$

Fonte: BORGES & CORDEIRO, (2012)

De forma análoga são calculadas as coordenadas (X_2,Y_2) verdadeiras do ponto P_2 que passa a ser chamado de P_{g2}, como ilustra a Figura B.2.

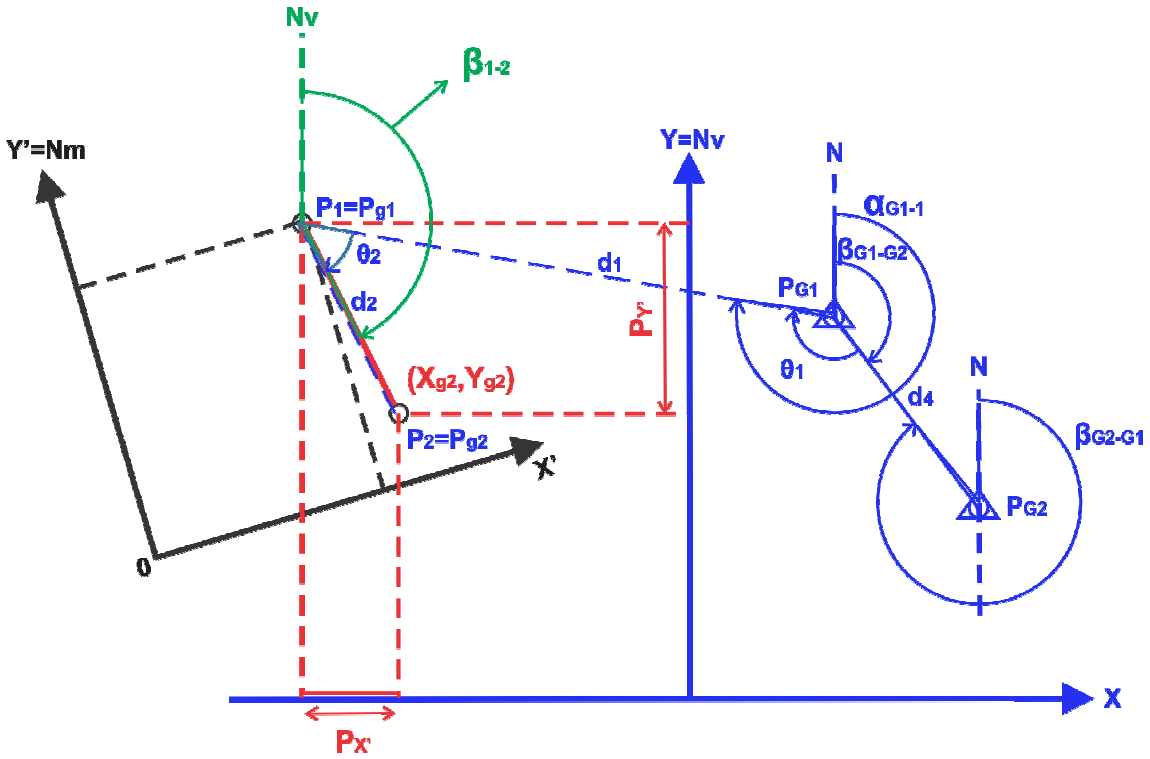


Figura B.2: Transporte das coordenadas do ponto P_2
Fonte: Autor

Para isto, aplicam-se os três passos para cálculo de azimute descrito anteriormente (equação B.7).

$$\beta_{1-2} = \begin{cases} se\ \alpha_{G1-1} + \theta_2 > 360 = (\alpha_{G1-1} + \theta_2) - 360 \\ se\ \alpha_{G1-1} + \theta_2 > 180 = (\alpha_{G1-1} + \theta_2) - 180 \\ se\ \alpha_{G1-1} + \theta_2 < 180 = (\alpha_{G1-1} + \theta_2) + 180 \end{cases} \quad \text{(B.7)}$$

Fonte: BORGES & CORDEIRO, (2012)

Com este azimute β_{1-2} e a distância d_2 (adquirida durante o levantamento não georreferenciado) calculam-se as projeções em *X* e em *Y* do alinhamento $P_{g1} - P_{g2}$ no sistema verdadeiro, da seguinte forma:

$$P_{x'} = d_2 \operatorname{sen}(\beta_{1-2}) \quad \text{(B.8)}$$

Fonte: BORGES & CORDEIRO, (2012)

$$P_{y'} = d_2 \cos(\beta_{1-2}) \quad \text{(B.9)}$$

Fonte: BORGES & CORDEIRO, (2012)

Com essas projeções calculam-se as coordenadas (X_2,Y_2) verdadeiras do ponto P_2, conforme segue:

$$X_2 = P_{x'} + X_{G2} = X_{g2} \quad \text{(B.10)}$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_2 = P_{y'} + Y_{G2} = Y_{g2} \quad \text{(B.11)}$$

Fonte: BORGES & CORDEIRO, (2012)

Os pontos P_{g1} e P_{g2} possuem a mesma localização física que os pontos P_1 e P_2 no terreno, só que possuem coordenadas verdadeiras que foram transportadas dos pontos georreferenciados P_{G1} e P_{G2} respectivamente, da forma demonstrada anteriormente. A denominação P_{g1} e P_{g2} para os pontos P_1 e P_2 diferencia os valores das coordenadas transportadas dos pontos georreferenciados P_{g1} e P_{g2}.

APÊNDICE C – Correção de erros em topografia e geodésia

Para ter o controle dos erros cometidos na medição é necessário medir a posição de um ponto de diferentes posições ou o que é mais usual, normalmente é formada uma poligonal dita enquadrada. Isto é, no caso da Figura C.1, medindo-se os ângulos A_1, A_2, A_3 e A_4 bem como calcular as distâncias d_1, d_2, d_3 e d_4 através das coordenadas georreferenciadas, forma-se uma poligonal enquadrada. Tendo em vista que os pontos P_{G1} e P_{G2} são pontos georreferenciados que pertencem a uma rede de triangulação geodésica do Sistema Geográfico Brasileiro e os pontos P_1 e P_2 pertencem a uma poligonal de um terreno levantado sem georreferenciamento e a poligonal do transporte de coordenadas está enquadrada entre as duas poligonais citadas, conforme Borges & Cordeiro (2012).

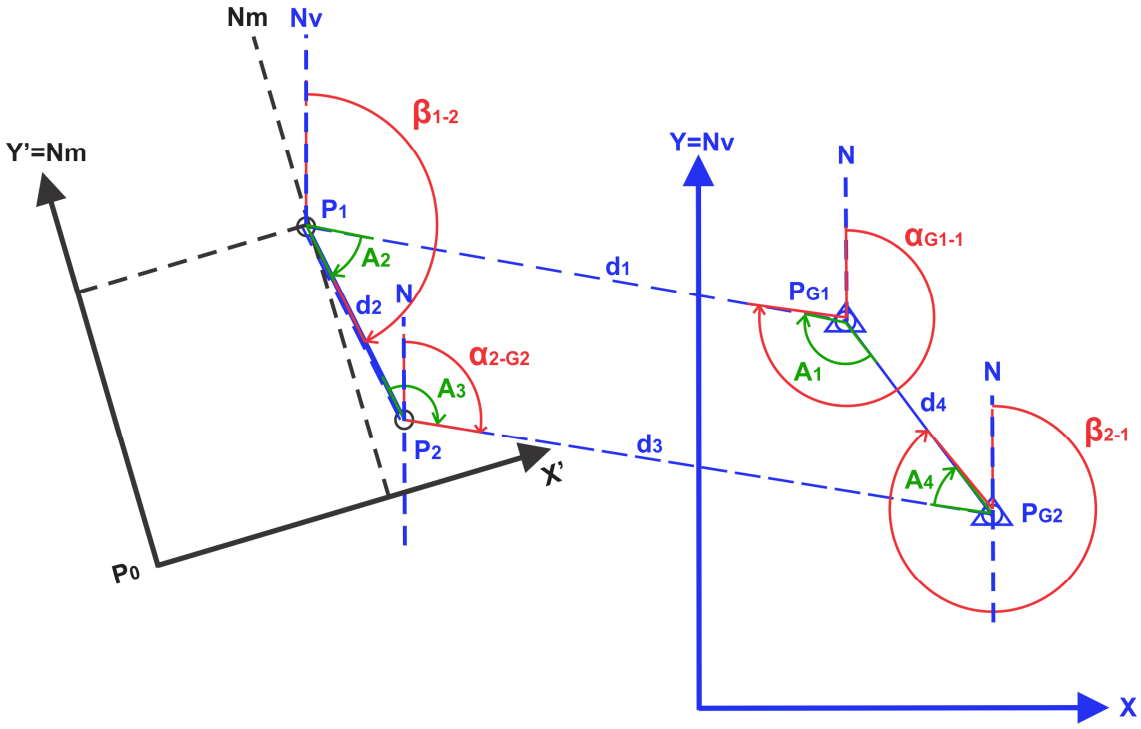


Figura C.1: Translação de coordenadas de sistemas diferentes
Fonte: Autor

No caso de uma poligonal fechada ou enquadrada, para que os erros sejam corrigidos, são executadas as ações a seguir, segundo Borges & Cordeiro (2012).

Primeiramente, deve-se verificar o fechamento angular da poligonal. Isto é, somar os ângulos internos ou externos da poligonal enquadrada levantada e comparar o resultado da soma com o resultado da equação da soma dos ângulos

internos (equação C.1) ou externos (equação C.2) de uma poligonal fechada qualquer, como abaixo:

$$\sum \alpha_i = 180\,(n-2) \tag{C.1}$$

Fonte: BORGES & CORDEIRO, (2012)

$$\sum \alpha_e = 180\,(n+2) \tag{C.2}$$

Fonte: BORGES & CORDEIRO, (2012)

Na Figura C.1, são utilizados ângulos internos em uma poligonal de quatro pontos (destacados em verde), na qual a soma desses ângulos (A_1 +A_2 + A_3 + A_4) deve ser igual a:

$$\sum \alpha_i = 180\,(4-2) = 360° \tag{C.3}$$

Fonte: BORGES & CORDEIRO, (2012)

Normalmente, a soma de tais ângulos será apenas próxima de 360º. Esta diferença é o erro cometido das medidas dos ângulos e deve estar de acordo com as tolerâncias permitidas. Essas tolerâncias normalmente são definidas como abaixo, conforme a finalidade do trabalho (áreas rurais, áreas urbanas e transportes geodésicos respectivamente):

$$T_l = 1' \sqrt{n} \tag{C.4}$$

Fonte: BORGES & CORDEIRO, (2012)

$$T_l = 30'' \sqrt{n} \tag{C.5}$$

Fonte: BORGES & CORDEIRO, (2012)

$$T_l = 10'' \sqrt{n} \tag{C.6}$$

Fonte: BORGES & CORDEIRO, (2012)

No caso de precisão geodésica a tolerância utilizada é calculada pela equação C.6, conforme a seguir:

$$T_l = 10'' \sqrt{4} = 20'' \tag{C.7}$$

Fonte: BORGES & CORDEIRO, (2012)

Ou seja, o erro máximo dos ângulos internos da poligonal não deve ultrapassar a margem de 20”. Por exemplo, se o resultado da soma dos ângulos for 360º 00’ 20’’, o erro seria de 20’’, o qual teria que ser corrigido dividindo o erro pelo número de ângulos e reajustando o resultado desta divisão em cada ângulo, fazendo com que a soma seja igual 360º.

Com os ângulos corrigidos calcula-se o azimute e as projeções em *X* e em *Y* de todos os alinhamentos da poligonal enquadrada, conforme descrito anteriormente na seção 2.5. Com as projeções calculadas, verifica-se o erro linear da seguinte forma:

$$\sum P_x = 0 \tag{C.8}$$

Fonte: BORGES & CORDEIRO, (2012)

$$\sum P_y = 0 \tag{C.9}$$

Fonte: BORGES & CORDEIRO, (2012)

Normalmente, a somatória das projeções em *X* e *Y* não serão igual a zero e sim próximas a esse valor, em função dos erros cometidos nas medidas das distâncias. Por exemplo, se a somatória em *X* for 0,04m e em *Y* for 0,03m (podendo ser maior ou menor a zero), então o erro em *X* (E_x) será igual a 0,04m e o erro em *Y* (E_y) será igual a 0,03m. Com isso, para calcular o erro total (E_T) utiliza-se a fórmula a seguir:

$$E_T = \sqrt{E_x^2 + E_y^2} \tag{C.10}$$

Fonte: BORGES & CORDEIRO, (2012)

Neste exemplo:

$$E_T = \sqrt{0{,}04^2 + 0{,}03^2} = 0{,}05m \tag{C.11}$$

Fonte: BORGES & CORDEIRO, (2012)

E as tolerâncias (que dependem sempre da finalidade do trabalho), quando não indicadas, podem ser usadas as seguintes:

$$T_l = \frac{\sum D}{2000} \quad \text{(C.12)}$$

Fonte: BORGES & CORDEIRO, (2012)

$$T_l = \frac{\sum D}{10000} \quad \text{(C.13)}$$

Fonte: BORGES & CORDEIRO, (2012)

Onde $\sum D$ é igual à soma dos lados da poligonal enquadrada. No exemplo, se a soma dos lados do terreno fosse igual a 570m, a tolerância para a área urbana seria:

$$T_l = \frac{570}{10000} = 0{,}057m \quad \text{(K.14)}$$

Fonte: BORGES & CORDEIRO, (2012)

Como o erro total cometido foi de 0,05m, então este estaria dentro da margem de erro.

O erro linear deve ser corrigido de forma proporcional à cada uma das distâncias seguindo os passos a seguir:

1) Calcula-se os fatores de correção das projeções em *X* e em *Y*:

$$F_{cx} = \frac{E_x}{\sum |P_x|} \quad \text{(C.15)}$$

Fonte: BORGES & CORDEIRO, (2012)

$$F_{cy} = \frac{E_y}{\sum |P_y|} \quad \text{(C.16)}$$

Fonte: BORGES & CORDEIRO, (2012)

2) Para corrigir as projeções em *X* e em *Y* deve-se multiplicar separadamente cada uma das projeções pelo seu respectivo fator de correção, obtendo-se dessa forma o valor da correção para cada uma das projeções, o qual deve ser somado ou subtraído (conforme o sinal do erro) do valor original de cada projeção. Conforme a seguir:

$$P_{xCorrigida} = \pm(P_x F_{cx}) + P_x \quad \text{(C.17)}$$

Fonte: BORGES & CORDEIRO, (2012)

$$P_{yCorrigida} = \pm(P_y F_{cy}) + P_y \quad \text{(C.18)}$$

Fonte: BORGES & CORDEIRO, (2012)

Com as projeções corrigidas é possível obter as coordenadas (*x,y*) de todos os pontos da poligonal enquadrada. Conforme a seguir, considerando dois pontos de uma poligonal que formam um alinhamento o qual resulta em projeções em *X* e em *Y* corrigidas:

$$X_{P(n+1)} = X_{P(n)} + P_{xCorrigida} \quad \text{(C.19)}$$

Fonte: BORGES & CORDEIRO, (2012)

$$Y_{P(n+1)} = Y_{P(n)} + P_{yCorrigida} \quad \text{(C.20)}$$

Fonte: BORGES & CORDEIRO, (2012)

Neste trabalho, durante a etapa de calibração são calculados os erros de medição resultantes das operações executadas pelo sistema, utilizando como base dois pontos localizados dentro do campo de visão da câmera e na área de trabalho do robô. Tais pontos possuem coordenadas georreferenciadas (fornecidas pelo robô) e também possuem coordenadas não georreferenciadas (fornecidas pela câmera) na mesma localização. Com isso, a poligonal enquadrada formada possui apenas três pontos (os dois pontos georreferenciados e a origem do sistema da câmera) que formam três lados, ou seja, um triângulo. Onde a origem do sistema da câmera (não georreferenciado), recebeu coordenadas transportadas do sistema georreferenciado, o que tornou os dois sistemas compatíveis.

Como os pontos utilizados na calibração já possuem coordenadas no sistema do robô, para realizar o cálculo do erro foi suficiente comparar as coordenadas transportadas no sistema do robô com as coordenadas do sistema do robô.

APÊNDICE D – Calibração da câmera

São descritos a seguir os passos executados neste trabalho para calibrar a câmera no programa *Vision Builder.*

- **Criar a calibração:** é necessário acessar a aba de calibração da etapa de aquisição de imagem e executar a opção de criar uma calibração, como ilustra a Figura D.1.

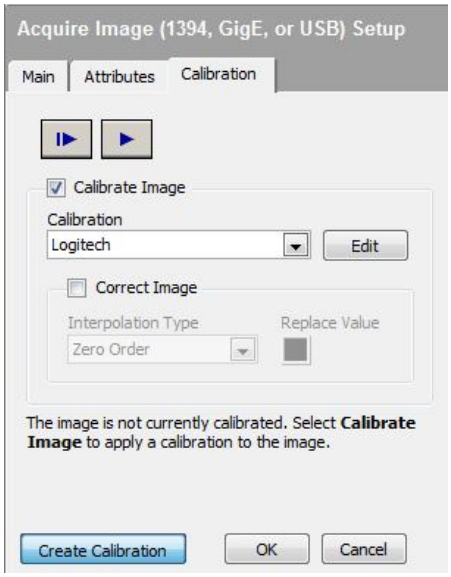


Figura D.1: Criação da calibração

Fonte: Autor

A primeira tela que aparece após selecionar esta opção é a ilustrada na Figura D.2, nela são configurados informações gerais da calibração, como o nome, o operador responsável e a validade.

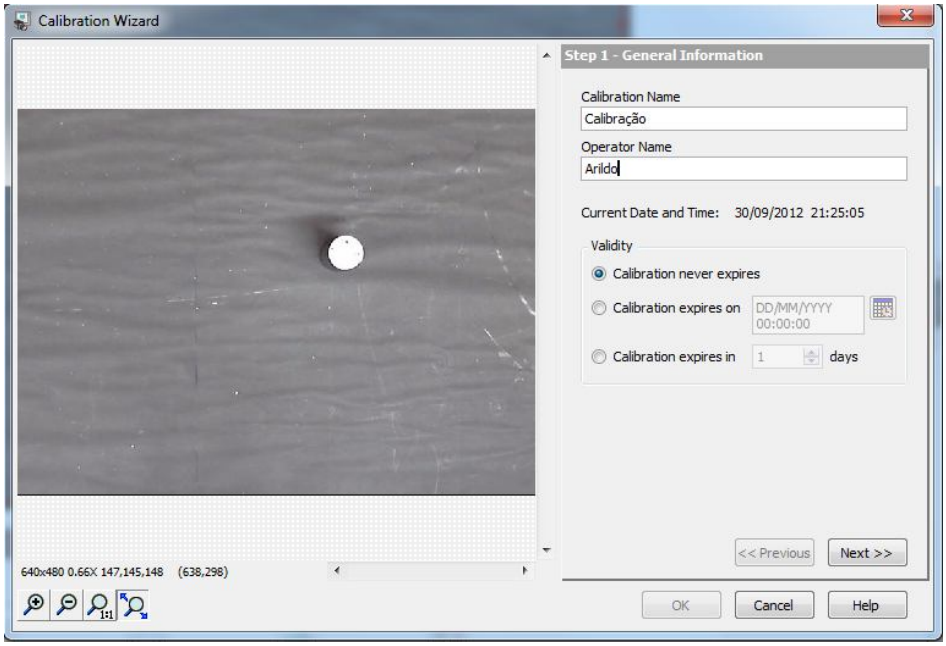


Figura D.2: Informações gerais da calibração

Fonte: Autor

- **Escolher o tipo de calibração:** existem três tipos de calibração disponíveis no *software Vision Builder*. O primeiro é a calibração simples, onde as coordenadas em *pixels* são transformadas em coordenadas do mundo real através da escalabilidade das direções em X e Y. O segundo é a calibração pelos pontos do usuário, isto é, o usuário informa diretamente ao *software* uma lista de pontos no mundo real e suas respectivas coordenadas no mundo real. E o terceiro é a calibração por grade, que consiste em definir em unidades do mundo real o espaçamento das direções em *X* e *Y* entre pontos de uma grade. A calibração utilizada neste trabalho é a simples, como mostra a Figura D.3, por ser um processo menos complexo que os demais tipos e por produzir o resultado esperado durante os testes, neste caso, o de calcular as dimensões reais dos objetos que aparecem na imagem e a distância real entre um ponto qualquer e a origem dos eixos coordenados da câmera.

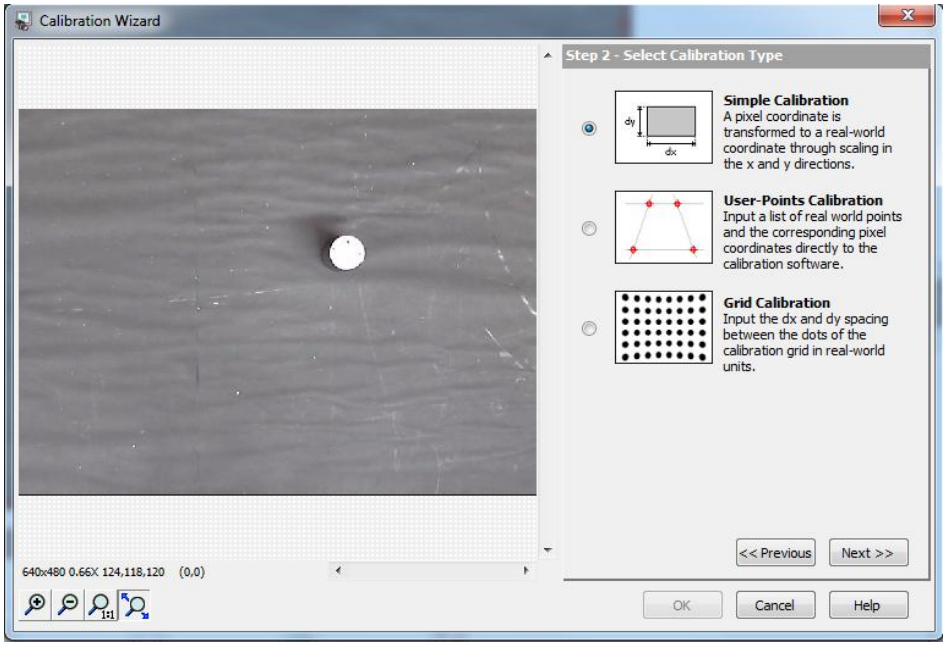


Figura D.3: Escolha do tipo de calibração

Fonte: Autor

- **Escolher uma imagem:** para que seja possível realizar a calibração, é necessário escolher uma imagem para ser utilizada durante todo o processo. Neste caso, optou-se por utilizar a última imagem adquirida pela *webcam*, como mostra a Figura D.4, porém também é possível utilizar uma imagem gravada no disco rígido do computador.

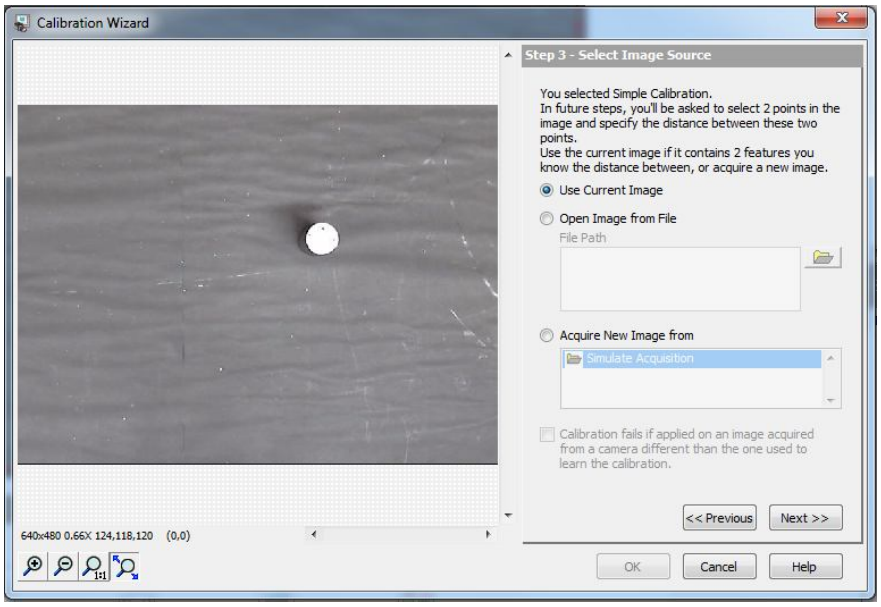


Figura D.4: Escolha da imagem a ser utilizada na calibração

Fonte: Autor

- **Escolher o tipo de pixel da câmera:** é necessário informar ao *software* se a câmera utiliza *pixels* quadráticos ou não, como ilustra a Figura D.5. Caso não utilize, será necessário especificar diferentes fatores de escalabilidade para os eixos coordenados *X* e *Y*. No caso da *webcam Logitech HD C270*, esta utiliza *pixels* quadráticos.

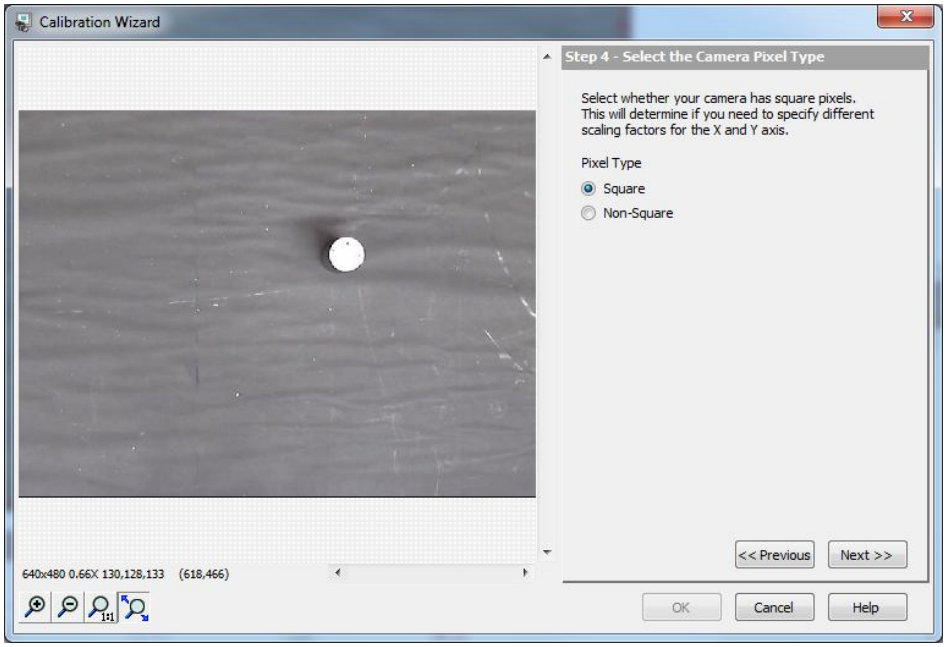


Figura D.5: Seleção do tipo de pixel da câmera

Fonte: Autor

- **Definir a relação dos *pixels* com o mundo real:** na calibração simples, são selecionados dois pontos quaisquer na imagem sendo utilizada e é informado ao *software* pelo usuário qual é a distância entre esses dois pontos em unidades do mundo real. Neste trabalho, estipularam-se dois pontos nas extremidades da base da peça, os quais determinam o seu diâmetro, cuja medida é conhecida no valor de 32 mm (milímetros), como mostra a Figura D.6.

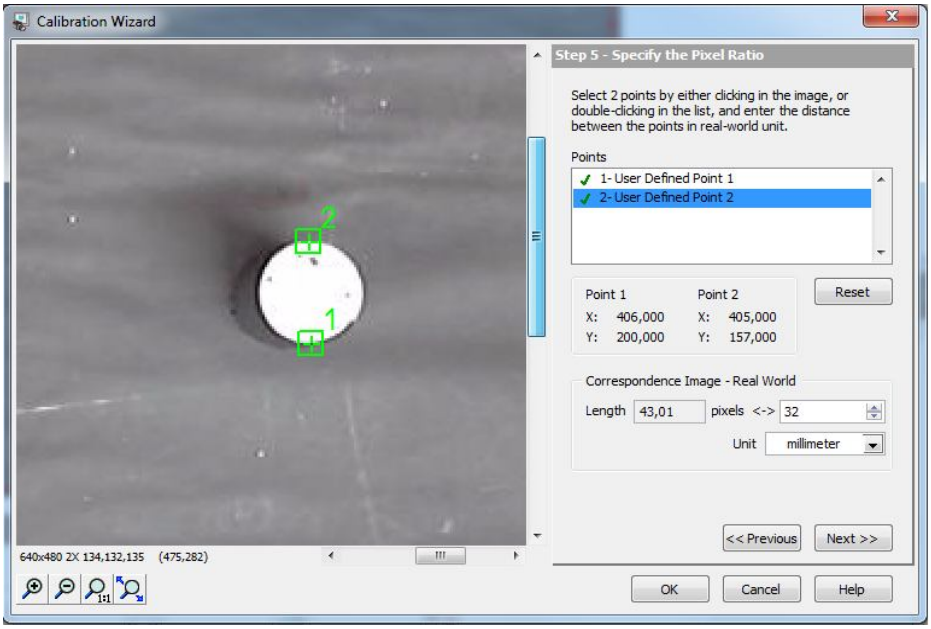


Figura D.6: Especificação da relação de *pixels* com o mundo real

Fonte: Autor

- **Definir a posição dos eixos coordenados:** é definida a posição da origem do sistema de coordenadas da câmera e sua respectiva orientação e angulação. Este passo é trivial para o correto funcionamento do sistema desenvolvido neste trabalho. A posição da origem deve ser no centro da imagem, caso contrário haverá distorções das medidas calculadas no mundo real, já que a câmera está posicionada paralela a mesa, o que garante que sua origem (o centro da sua visão) está localizada no centro da imagem adquirida. A orientação dos eixos coordenados deve ser a mesma utilizada pelo sistema de coordenadas do robô, senão a relação entre os dois sistemas (do robô e da câmera) poderá estar invertida. E a angulação dos eixos deve apenas estar a 0°, pois desta forma irá garantir que após sua rotação ser realizada pelo programa de manipulação, os sistemas de coordenadas do robô e da câmera estarão paralelos. A definição da posição dos eixos coordenados é ilustrada na Figura D.7.

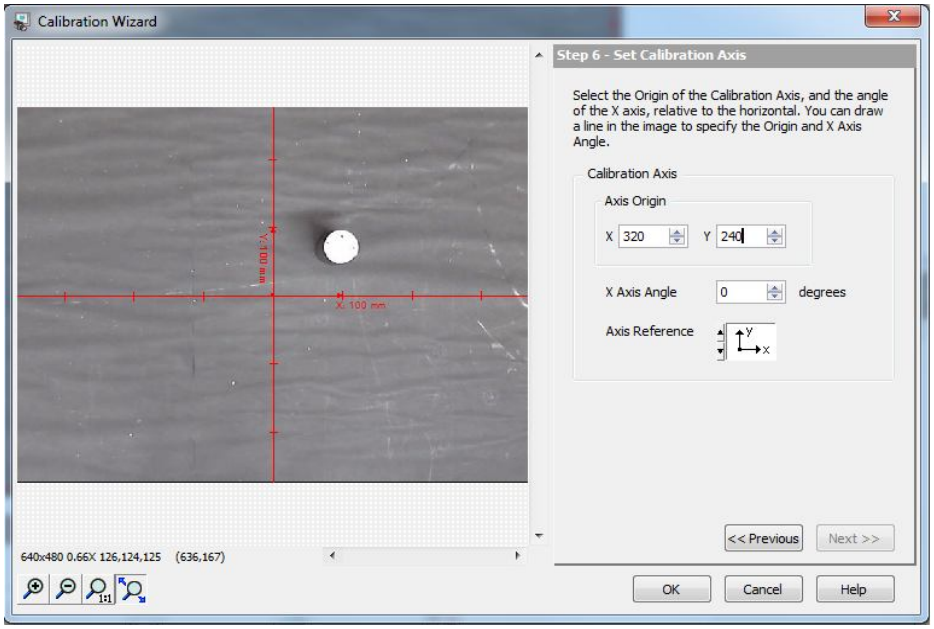


Figura D.7: Definição da posição dos eixos coordenados

Fonte: Autor

APÊNDICE E – Comandos do robô RV-M1

Comando	Descrição	Formato
DP *(Decrement Position)*	Move o robô para posição anterior a atual	DP
DW *(Draw)*	Move o final da mão para uma posição distante da atual respeitando as distâncias especificadas em cada eixo	DW <DistânciaX>, <DistânciaY>, <DistânciaZ>
HE *(Here)*	Grava a posição atual do robô (coordenadas) em uma posição numérica	HE <Número da Posição>
HO *(Home)*	Estabelece uma posição de referência no plano de coordenadas cartesianas	HO
IP *(Increment Position)*	Move o robô para a posição seguinte a atual	IP
MA *(Move Approach)*	Move o final da mão para uma posição distante da atual respeitando os incrementos especificados entre duas posições indicadas	MA <Posição número (a)>, <Posição número (b)>, <O ou C>
MC *(Move Continuous)*	Move o robô continuamente entre dois pontos predefinidos	MC <Posição número (a)>, <Posição número (b)>
MJ *(Move Joint)*	Movimenta cada junta de acordo com os ângulos especificados a partir da posição atual.	MJ <Ângulo da cintura>, <Ângulo do ombro>, <Ângulo do cotovelo>, <Ângulo do pulso>, <Ângulo de giro>
MO *(Move)*	Move o final da mão para uma posição especificada	MO<Número da Posição>, <O ou C>
MP *(Move Position)*	Move o final da mão para uma posição especificada a partir de coordenadas e ângulos	MP <Coordenada X>, <Coordenada Y>, <Coordenada Z>, <Ângulo do pulso>, <Ângulo de giro>
MS *(Move Straight)*	Move o robô para uma posição especificada com a quantidade de pontos intermediários que devem ser seguidos em linha reta	MS <Número da posição>, <Quantidade de pontos intermediários>, <O ou C> A quantidade de pontos deve ser entre 1 e 99
MT *(Move Tool)*	Move o final da mão em linha reta a partir da posição atual com uma distancia predefinida	MT <Número da posição>, <Distância>, <C ou O>
NT *(Nest)*	Retorna o robô para sua posição de origem mecânica	NT
OG *(Origin)*	Move o robô para a posição de origem do plano cartesiano	OG
PL *(Position Load)*	Atribui as coordenadas da posição B à posição A.	PL <Número da posição A>, <Número da posição B>

PX *(Position Exchange)*	Troca as coordenadas da posição B com as da posição A	PX <Número da posição A>, <Número da posição B>
SF *(Shift)*	Adiciona as coordenadas da posição B às coordenadas da posição A	SF <Posição A>, <Posição B>
SP *(Speed)*	Define a velocidade, aceleração (H) ou desaceleração (L) na aproximação do ponto de destino	SP <Velocidade>, <H ou L> Onde velocidade varia entre 0 e 9.
TI *(Timer)*	Para o movimento do robô por um período de tempo determinado	TI <Tempo> Onde tempo pode variar de 0.1 a 32767 segundos
TL *(Tool)*	Estabelece uma distancia entre o final do braço robótico e a superfície que será trabalhada (tamanho de ferramenta)	TL <Distância> Onde a distância pode variar entre 0 e 300 mm. O padrão é 0.
DL *(Delete Line)*	Deleta o conteúdo de um determinado intervalo de linhas	DL <Linha A>, <Linha B> Deleta inclusive o conteúdo das linhas A e B.
GC (Grip Close)	Fecha e mantém a garra fechada	GC
GO (Grip Open)	Abre e mantém a garra aberta	GO
ED *(End)*	Encerra o programa	ED
GT (*Go To*)	Envia o robô para uma linha especifica do programa (não condicional)	GT <Linha do programa>
ER (*Error Read*)	Lê o estado de erro usando RS-232	ER
LR (*Line Read*)	Lê a linha especificada do programa	LR <Linha do programa>
PR (*Position Read*)	Lê as coordenadas de uma posição especifica do robô	PR <Número da posição>
WH (*Where*)	Lê as coordenadas da posição atual do robô	WH
EQ *(If Equal)*	Compara um valor de um registro interno com um valor especificado e se igual vai para uma linha especificada de programa	EQ <Valor a ser comparado>, <Linha do programa>
NE (*If Not Equal*)	Compara um valor de um registro interno com um valor especificado e se não igual vai para uma linha especificada de programa	NE <Valor a ser comparado>, <Linha do programa>
RC (*Repeat Cycle*)	Repete um ciclo (delimitado pelo comando NX) por um número de vezes especificado	RC <Número de repetições>
NX (*Next*)	Especifica o final do ciclo de repetição iniciado pelo comando RC	NX

RN (*Run*)	Executa as linhas do programa de linha A até B (ou até ED)	RN <Linha A>, <Linha B>
RS (*Reset*)	Reseta o programa e as condições de erro	RS
´ (*Coment*)	Adiciona comentário ao programa	´ <Comentário>

Fonte: MALAQUIAS & ASSOLAR & NOGAWA, (2012)

APÊNDICE F – Conceitos de robótica para realizar a rotação e translação de sistemas

Segundo Craig (2012), a rotação e translação dos sistemas apresentados neste trabalho podem ser calculadas utilizando os conceitos de robótica de descrições espaciais e transformações.

Inicialmente, é realizado um processo de calibração utilizando dois pontos de coordenadas conhecidas no sistema do robô e da câmera. Isto é, os passos de 1 a 4 apresentados na seção de calibração 3.5.2 são executados de forma a adquirir as coordenadas dos dois pontos.

Com isso, os pontos P_1 e P_2 são representados através de vetores posição no sistema do robô e da câmera, como mostram as equações F.1 e F.2:

$$ {}^{R}P_1 = \begin{bmatrix} {}^{R}x_1 \\ {}^{R}y_1 \\ {}^{R}z_1 \\ 1 \end{bmatrix} \qquad {}^{R}P_2 = \begin{bmatrix} {}^{R}x_2 \\ {}^{R}y_2 \\ {}^{R}z_2 \\ 1 \end{bmatrix} \tag{F.1} $$

Equação F.1: Vetores posição dos ponto P_1 e P_2 no sistema do robô

$$ {}^{C}P_1 = \begin{bmatrix} {}^{C}x_1 \\ {}^{C}y_1 \\ {}^{C}z_1 \\ 1 \end{bmatrix} \qquad {}^{C}P_2 = \begin{bmatrix} {}^{C}x_2 \\ {}^{C}y_2 \\ {}^{C}z_2 \\ 1 \end{bmatrix} \tag{F.2} $$

Equação F.2: Vetores posição dos ponto P_1 e P_2 no sistema da câmera

Com os vetores-posição das equações F.1 e F.2, segundo Craig (2012) calculam-se a rotação e translação entre os sistemas da câmera e do robô através da equação F.3 abaixo:

$$ {}^{R}P = {}^{R}T_C \, {}^{C}P \tag{F.3} $$

Equação F.3: Transformação do sistema da câmera para o sistema do robô

Fonte: CRAIG, (2012)

Onde:

${}^{R}P =$ matriz dos vetores posição dos pontos P_1 e P_2 no sistema do robô

${}^{R}T_C =$ matriz transformada do sistema da câmera para o sistema do robô

${}^{C}P =$ matriz dos vetores posição dos pontos P_1 e P_2 no sistema da câmera

A equação F.3 em forma de operadores matriciais é apresentada na equação F.4. Os parâmetros em z dos vetores foram substituídos pelo valor zero, considerando que este trabalho possui a coordenada z fixa.

$$\begin{bmatrix} {}^{R}x_1 & {}^{R}x_2 \\ {}^{R}y_1 & {}^{R}y_2 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\text{sen}\,\theta & 0 & x_{Corg} \\ \text{sen}\,\theta & \cos\theta & 0 & y_{Corg} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{C}x_1 & {}^{C}x_2 \\ {}^{C}y_1 & {}^{C}y_2 \\ 0 & 0 \\ 1 & 1 \end{bmatrix} \quad \text{(F.4)}$$

Fonte: CRAIG, (2012)

A evolução dos cálculos da equação F.4 são representados pelas equações F.5 a F.8. Os resultados de tais equações fornecem o ângulo de rotação θ e as coordenadas da origem do sistema da câmera no sistema do robô (x_{Corg}, y_{Corg}).

$${}^{R}x_1 = {}^{C}x_1 \cos\theta - {}^{C}y_1 \,\text{sen}\,\theta + x_{Corg} \quad \text{(F.5)}$$

$${}^{R}y_1 = {}^{C}x_1 \,\text{sen}\,\theta + {}^{C}y_1 \cos\theta + y_{Corg} \quad \text{(F.6)}$$

$${}^{R}x_2 = {}^{C}x_2 \cos\theta - {}^{C}y_2 \,\text{sen}\,\theta + x_{Corg} \quad \text{(F.7)}$$

$${}^{R}y_2 = {}^{C}x_2 \,\text{sen}\,\theta - {}^{C}y_2 \cos\theta + y_{Corg} \quad \text{(F.8)}$$

Na seção 3.5.2 foram utilizados conceitos de topografia e geodésia para realizar os mesmos cálculos. O ângulo de rotação θ foi calculado no passo 6 através da diferença entre os azimutes (α_{1-2}, β_{1-2}) do alinhamento entre os pontos P_1 e P_2 no sistema da câmera e do robô (equação F.9). Já a translação, isto é, calcular as coordenadas da origem do sistema da câmera em relação ao sistema do robô (x_{Corg}, y_{Corg}), foi calculada nos passos 7 a 9 utilizando as projeções rotacionadas e as coordenadas no sistema do robô para o ponto P_1 (equações F.10 e F.11).

$$\theta = \alpha_{1-2} - \beta_{1-2} \quad \text{(F.9)}$$

Equação F.9: Cálculo do ângulo de rotação utilizando conceitos de topografia e geodésia

Fonte: BORGES & CORDEIRO, (2012)

$$x_{Corg} = P_{Xr'} + X_{g1} \quad \text{(F.10)}$$

Equação F.10: Cálculo da coordenada x_{Corg} utilizando conceitos de topografia e geodésia

Fonte: BORGES & CORDEIRO, (2012)

$$y_{Corg} = P_{Yr'} + Y_{g1} \quad \text{(F.11)}$$

Equação F.11: Cálculo da coordenada y_{Corg} utilizando conceitos de topografia e geodésia

Fonte: BORGES & CORDEIRO, (2012)

Onde:

$\theta =$ equivale à rotação θ da matriz transformada ${}^{R}T_{C}$

$x_{Corg} =$ equivale à coordenada x_{Corg} da matriz transformada ${}^{R}T_{C}$

$y_{Corg} =$ equivale à coordenada y_{Corg} da matriz transformada ${}^{R}T_{C}$

$X_{g1} =$ equivale à coordenada ${}^{R}x_{1}$

$Y_{g1} =$ equivale à coordenada ${}^{R}y_{1}$

$P_{Xr'} =$ equivale à projeção rotacionada (no sistema do robô) da coordenada ${}^{C}x_{1}$

$P_{Yr'} =$ equivale à projeção rotacionada (no sistema do robô) da coordenada ${}^{C}y_{1}$

Realizada a rotação e translação entre os sistemas através das equações F.5 a F.8, é possível calcular as coordenadas para um ponto desconhecido P_n do sistema da câmera em relação ao sistema do robô através da equação F.12.

$$\begin{bmatrix} {}^{R}x_n \\ {}^{R}y_n \end{bmatrix} = \begin{bmatrix} \cos\theta & \text{sen}\,\theta \\ -\text{sen}\,\theta & \cos\theta \end{bmatrix} \begin{bmatrix} {}^{C}x_n \\ {}^{C}x_n \end{bmatrix} + \begin{bmatrix} x_{Corg} \\ y_{Corg} \end{bmatrix} \quad \text{(F.12)}$$

Equação F.12: Cálculo das coordenadas para um ponto desconhecido P_n

Em topografia e geodésia, na seção 3.5.3, as coordenadas do ponto P_n são calculadas nos passos 1 a 6 utilizando as projeções rotacionadas e as coordenadas da origem do sistema da câmera em relação ao sistema do robô (x_{Corg}, y_{Corg}) como mostrado nas equações F.13 e F.14.

$$X_n = P_{Xr'} + x_{Corg} \quad \text{(F.13)}$$

Equação F.13: Cálculo da coordenada X_n utilizando conceitos de topografia e geodésia

Fonte: BORGES & CORDEIRO, (2012)

$$Y_n = P_{Yr'} + y_{Corg} \quad \text{(F.14)}$$

Equação F.14: Cálculo da coordenada Y_n utilizando conceitos de topografia e geodésia

Fonte: BORGES & CORDEIRO, (2012)

Onde:

X_n = equivale à coordenada ${}^{R}x_n$

Y_n = equivale à coordenada ${}^{R}y_n$

$P_{Xr'}$ = equivale à projeção rotacionada (no sistema do robô) da coordenada ${}^{C}x_n$

$P_{Yr'}$ = equivale à projeção rotacionada (no sistema do robô) da coordenada ${}^{C}y_n$

x_{Corg} = equivale à coordenada x_{Corg} da matriz transformada ${}^{R}T_C$

y_{Corg} = equivale à coordenada y_{Corg} da matriz transformada ${}^{R}T_C$