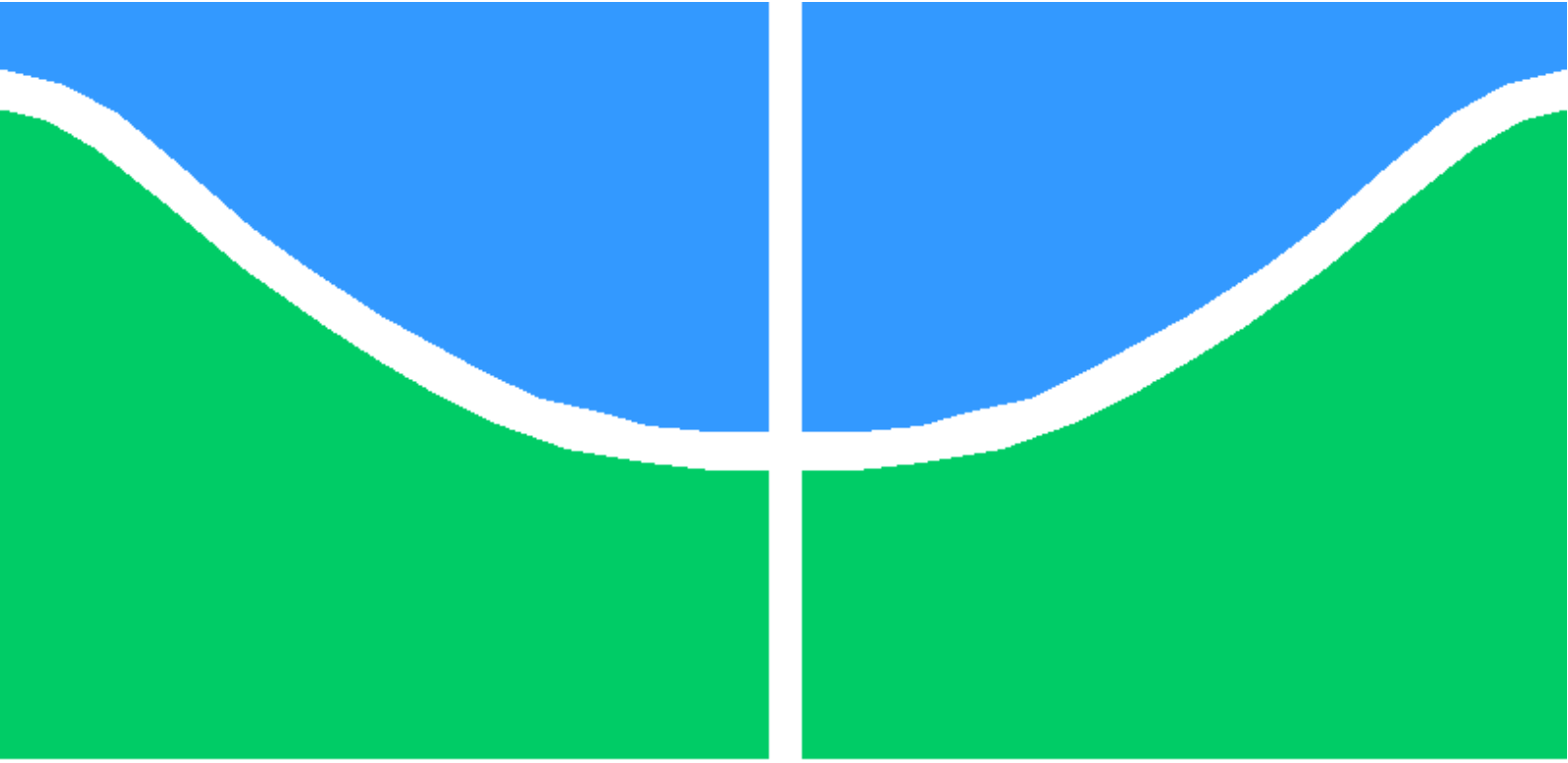


TRABALHO DE GRADUAÇÃO

SISTEMA DE LOCALIZAÇÃO PARA ROBÓTICA MÓVEL COM RFID

André Luiz Gama de Souza

Gabriel Figueiró de Oliveira

Brasília, Julho de 2011

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

SISTEMA DE LOCALIZAÇÃO PARA ROBÓTICA MÓVEL COM RFID

André Luiz Gama de Souza

Gabriel Figueiró de Oliveira

Relatório submetido como requisito parcial para obtenção do grau de Engenheiro de Controle e Automação

Banca Examinadora

Prof. Geovany Araújo Borges, ENE/UnB _____
Orientador

Prof. Adolfo Bauchspiess, ENE/UnB _____
Co-orientador

Prof. Renato Alves Borges, ENE/UnB _____
Examinador interno

FICHA CATALOGRÁFICA

SOUZA, ANDRÉ LUIZ GAMA DE; OLIVEIRA, GABRIEL FIGUEIRÓ DE
SISTEMA DE LOCALIZAÇÃO PARA ROBÓTICA MÓVEL COM RFID [Distrito Federal] 2011.
xi, 92p., 210 x 297 mm (FT/UnB, Engenheiro de Controle e Automação, 2011).
Graduação - Universidade de Brasília, Faculdade de Tecnologia.

1. Robótica Móvel
2. RFID
3. Realidade aumentada
4. Filtragem estocástica

I. Engenharia Mecatrônica/FT/UnB

REFERÊNCIA BIBLIOGRÁFICA

SOUZA, A.L.G.; OLIVEIRA, G.F. (2011). SISTEMA DE LOCALIZAÇÃO PARA ROBÓTICA MÓVEL COM RFID, Trabalho de Graduação, Publicação FT.TG-x/11, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 92p.

CESSÃO DE DIREITOS

AUTOR: André Luiz Gama de Souza; Gabriel Figueiró de Oliveira
TÍTULO: SISTEMA DE LOCALIZAÇÃO PARA ROBÓTICA MÓVEL COM RFID.
GRAU: Engenheiro de Controle e Automação ANO: 2011

Dedicatórias

À minha família e a todos que de alguma forma ajudaram e tornaram possível essa realização.

Gabriel Figueiró de Oliveira

Dedico este trabalho ao meu pai, Álvaro, à minha mãe, Júlia, e à minha namorada, Débora.

André Luiz Gama de Souza

Agradecimentos

A gradeço à minha família, com ênfase em meus pais, por tudo que me ensinaram; à minha namorada, que, de forma paciente, esteve ao meu lado durante o curso; aos professores que participaram da minha formação, em especial meus orientadores, Geovany e Adolfo; aos meus amigos do LARA, sempre presentes em bons e maus momentos acadêmicos; e aos meus amigos do bar (agora acreditam que eu estava ocupado?). Por fim, um agradecimento especial a todos que contribuíram de alguma forma para esse trabalho, não citando nomes pelo medo de esquecimento.

André Luiz Gama de Souza

Sem os valores que me foram passados desde sempre, seria impossível chegar até aqui pelo caminho que escolhi trilhar. Por isso em primeiro lugar agradeço à minha família, mesmo tendo ouvido tantas vezes conselhos para abandonar a engenharia e fazer um concurso público. Agradeço à minha namorada, por ter me acompanhado e demonstrado tanto amor e paciência durante toda esta etapa tão importante. Por último, agradeço aos bons amigos, que dividiram momentos de angústia e de realização, engrandecendo a minha experiência na universidade, incluindo aqui, os meus orientadores.

Gabriel Figueiró de Oliveira

RESUMO

O presente trabalho trata do estudo e desenvolvimento de sistemas de localização para robótica móvel, visando uma contribuição com o projeto de rastreamento de *tags* RFID (Identificação por Rádio Frequência) do Laboratório de Robótica e Automação da Universidade de Brasília. No decorrer do desenvolvimento são utilizadas técnicas de estimação e filtragem estocástica e operações com sistemas de coordenadas que são detalhadas na fundamentação teórica do trabalho. É proposto um sistema de localização utilizando realidade aumentada e um sistema com uso de RFID. A instrumentação para realização de experimentos conforme necessário ao projeto de rastreamento com RFID é desenvolvida alcançando resultados satisfatórios. São realizados ensaios com RFID gerando arquiteturas de soluções para o projeto de localização em questão.

ABSTRACT

This work aims to study and develop an indoor localization system for mobile robots, as a contribution to the RFID (Radio Frequency Identification) tags tracking project of the Laboratory of Automation and Robotics at the University of Brasília. In the course of development, several techniques are used, including coordinate system transforms, stochastic filtering and estimation techniques, which are detailed throughout the text. A localization system using augmented reality and a system using RFID are proposed. A framework for experimentation was developed for the RFID tracking project, achieving satisfactory results. Tests were conducted, leading to RFID solution architectures for the localization project in question.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Símbolos

X	Eixo coordenado	
Y	Eixo coordenado	
Z	Eixo coordenado	
χ	Pontos sigma gerados na transformação unscented	
κ	Número referente à dispersão de pontos sigma	
α	fator de atenuação do sinal de RSSI em ambientes internos	
$\mathbf{x}$	Vetor de estados do Filtro de Kalman	
$\mathbf{z}$	Medição do Filtro de Kalman	
Q	Matriz de covariância associada ao passo de predição do filtro de Kalman	
R	Matriz de covariância associada ao passo de medição do filtro de Kalman	
P	Matriz de covariância associada ao vetor de estados do filtro de Kalman	
S	Matriz de covariância associada a dispersão da estimativa da medida no FKU	
u	entrada do Filtro de Kalman	
K	Ganho de Kalman calculado na fase de correção do Filtro de Kalman	
l	Distância entre roda e centro do robô	[mm]
b	Distância entre as rodas do robô	[mm]
v	Velocidade linear do robô	[mm/s]
r	Raio da roda do robô	[mm]
ω_l	Velocidade angular da roda esquerda do robô	[rad/s]
ω_r	Velocidade angular da roda direita do robô	[rad/s]
v_l	Velocidade de linear da roda esquerda do robô	[mm/s]
v_r	Velocidade de linear da roda direita do robô	[mm/s]
Γ	Desvio linear da trajetória do robô	[mm]
Θ	Desvio angular da trajetória do robô	[rad]
K_Γ	Ganho do controlador proporcional referente ao desvio linear da trajetória	
K_Θ	Ganho do controlador proporcional referente ao desvio angular da trajetória	

Subscritos

k	k-ésima amostragem

Sobrescritos

$\bar{}$	Valor médio da variável
$\hat{}$	Valor estimado da variável

Siglas

AR	*Augmented Reality* (Realidade Aumentada)
GPS	*Global Positioning System* (Sistema de Posicionamento Global)
FDP	Função de Densidade de Probabilidade
FK	Filtro de Kalman
FKE	Filtro de Kalman Estendido
FKU	Filtro de Kalman *Unscented*
HMD	*Head Mounted Display* (Display montado na cabeça)
LARA	Laboratório de Automação e Robótica
MMQ	Método dos Mínimos Quadrados
RF	Rádio Frequência
RFID	*Radio Frequency Identification* (Identificação por Rádio Frequência)
RNA	Rede Neural Artificial
Rot	Rotação
RSSI	*Received Signal Strength Indicator* (Indicador de Força do Sinal Recebido)
Trans	Translação
TU	Transformação *Unscented*
UnB	Universidade de Brasília

Capítulo 1

Introdução

Neste capítulo é apresentada a principal motivação para o projeto, localização em ambientes internos, relatando um breve histórico de trabalhos relacionados. Aqui são apresentados os objetivos do projeto, explicitando o contexto no qual está inserido. Por fim, é feita uma breve descrição dos capítulos seguintes.

1.1 Contextualização

A necessidade de conhecer o posicionamento de pessoas ou até mesmo objetos sempre esteve presente nas atividades humanas. Nos primórdios da civilização, o homem formulou teorias sobre a sua localização baseado em observações celestes. Felizmente, hoje em dia existem métodos menos rudimentares disponíveis, embora estejam, muitas vezes, baseados nos mesmos princípios. O Sistema de Posicionamento Global, GPS (do inglês, *Global Positioning System*), é uma das ferramentas mais utilizadas atualmente. O sistema é usado para as mais diversas aplicações, incluindo jogos, aplicativos para celulares e localização veicular.

O GPS consiste em 24 satélites percorrendo diferentes órbitas em torno da Terra, de maneira que sempre haja pelo menos cinco deles visíveis de qualquer lugar do planeta [3]. Estações de monitoramento e antenas ao redor do mundo adquirem dados dos satélites de forma passiva, enviando-os a uma base central para processamento.

Devido a diversos fatores, é comum que haja um erro no sistema de GPS padrão, o qual funciona por triangulação simples, utilizando a potência do sinal recebido para cálculo da distância. De acordo com [4], é possível melhorar o resultado do GPS com algumas técnicas. A principal utilizada é a correção do erro de posicionamento por meio de bases fixas, permitindo uma melhor estimação da posição do satélite. Outra possibilidade é a fusão sensorial, por exemplo, com uma unidade inercial. O uso de algoritmos, como o Filtro de Kalman (FK), que calibram recursivamente os parâmetros do GPS, também auxiliam na estimação. Outras técnicas mais complexas também são possíveis, como conhecer restrições do usuário para estimativa do sinal, ao saber sua velocidade e posição em instantes passados.

Apesar do GPS obter resultados excelentes em ambientes externos, ele não é muito eficiente em ambientes internos, onde não se tem uma linha de visada direta entre receptor e satélite. Segundo [5], receptores GPS demoram entre um e dois minutos para encontrar satélites, com a restrição de que o sinal não pode estar muito atenuado. Nesse contexto, esforços são realizados para possibilitar a localização em ambientes internos.

Diversas tecnologias vêm sendo empregadas para a localização em ambientes internos. No projeto *LANDMARC* [6], um dos primeiros trabalhos relatados para localização com RFID (do inglês *Radio Frequency Identification* - Identificação por Rádio Frequência), os autores utilizam a tecnologia para determinar o posicionamento de *tags*. Ainda no trabalho, os autores apresentam um estudo sobre outras tecnologias, sendo algumas delas listadas a seguir:

- Infravermelho: a necessidade de estar em linha de visada e o seu sinal de curto alcance fizeram com que tal técnica fosse bem limitada.
- *IEEE 802.11*: sistema baseado em Rádio Frequência (RF), o qual utiliza o sinal recebido de diversos emissores de posição conhecida. Requer uma infraestrutura simples, porém o sinal ruidoso não permite uma localização acurada.
- Ultra-som: sensor baseado em sistemas de localização de animais como o morcego. Consegue obter um resultado satisfatório, porém, requer uma estrutura física que muitas vezes não é viável.

O RFID, apesar de não ter sido concebido para esse propósito, tem sido utilizado para realizar a localização em ambientes internos. Trabalhos como [7, 8] propuseram modificações no *LANDMARC*, resultando em uma leve melhora. Os resultados utilizando o RFID de forma determinística para a localização foram bons em ambientes controlados, geralmente não sendo apresentadas as conclusões para outros tipos de ambientes. Alguns trabalhos, como [9, 10], utilizam uma abordagem probabilística. Essa linha de pensamento encontra respaldo nas características do sinal de RFID como sensor de distância, pois, sendo muito ruidoso, deve ter uma incerteza associada à sua medida.

Na Universidade de Brasília (UnB), o Laboratório de Robótica e Automação (LARA) começou a desenvolver pesquisas utilizando RF para fazer localização em ambientes internos. Em [11], os autores utilizam a potência do sinal recebido por uma rede de sensores *ZigBee* para localizar um objeto. Seguindo a mesma linha, no trabalho [12], o autor treina uma rede neural para interpolar dados de potência de sinal recebido por leitoras RFID, visando utilizar a informação de localização das pessoas para racionalização de energia.

A idéia de localização utilizando RFID, no LARA, inicialmente concebida para ambientes inteligentes, despertou o interesse da área de robótica móvel, para ser utilizada como um sensor de localização. Com um sensor a mais, é possível realizar uma fusão sensorial para melhorar a estimativa do posicionamento do robô. Em contrapartida, um modelo de evolução da posição (obtido em robôs móveis a partir de um modelo cinemático) pode facilitar a tarefa de localização, dando assim base a um posterior modelamento de evolução da posição de indivíduos.

1.2 Definição do problema

O projeto de localização com RFID está em fase inicial, e ainda necessita ser melhor estruturado. No momento, as únicas informações relevantes são referentes ao trabalho [12], que buscou caracterizar, para ambientes internos, a relação entre a força do sinal RFID recebido e a distância entre o emissor e o receptor.

Uma necessidade básica é a estrutura de aquisição de dados no robô móvel, que deve ser feita com um relógio sincronizado. Considerando o sistema de localização RFID para robótica, é necessário um sistema de localização auxiliar com alto nível de confiança, permitindo avaliar os resultados da localização com RFID. Além disso, o robô precisa ser capaz de seguir uma trajetória, permitindo a repetição de experimentos. Também é necessário um modelo de propagação de sinal RFID para ambiente interno, para que a informação do seu sinal seja utilizada.

1.3 Objetivos do projeto

O presente trabalho de graduação teve como principal objetivo contribuir com o projeto de localização em ambientes internos utilizando RFID. Além do cálculo de posicionamento propriamente dito, é necessário testar a sua confiabilidade. Atualmente, não é possível comparar o resultado obtido com a posição real da *tag* quando em movimento. Utilizando um robô móvel para portar a *tag* que se deseja localizar, permite-se relacionar a posição correta do robô com os dados de RFID adquiridos. Também é desejado que os experimentos sejam repetidos de forma mais exata com o robô móvel, viabilizando a execução em diversas condições.

Foi desenvolvido um sistema de localização com câmera para o robô móvel utilizando o Filtro de Kalman *Unscented*. Dessa forma, sua posição no ambiente é calculada de maneira satisfatória, esteja em repouso ou em movimento. Assim, é possível utilizar o resultado do posicionamento para comparar com o desempenho de outros sistemas de localização, incluindo outros trabalhos que não RFID. A tarefa de localização deve ter o relógio sincronizado com a aquisição de dados das leitoras RFID.

Para fosse possível repetir a passagem da *tag* por pontos fixos no ambiente, foi desenvolvido um controle de trajetória embarcado no robô, permitindo que caminhos semelhantes sejam testados em diversas condições. Dessa forma, é possível programar o robô para que passe por pontos determinados e conhecidos, minimizando o erro caso seja comandado por controle remoto, facilitando a repetição de experimentos.

Por fim, foi feita uma proposta inicial para que seja desenvolvida a localização utilizando RFID. Os testes do sistema proposto foram iniciados, permitindo algumas conclusões relevantes e satisfatórias.

1.4 Apresentação do manuscrito

No capítulo 2 são expostos alguns temas teóricos de maior relevância para o desenvolvimento do trabalho. Inicialmente é apresentado um estudo sobre transformações de coordenadas cartesianas. Em seguida, o Filtro de Kalman é explicado, assim como suas variantes. Por fim, é detalhado sobre o controlador de trajetória desenvolvido no trabalho.

O capítulo 3 descreve os temas práticos. Nesse capítulo será apresentado o desenvolvimento do trabalho, convenções e ferramentas adotadas. Aqui também serão apresentados os estudos desenvolvidos e os experimentos realizados, descrevendo seus objetivos, suas realizações e seus resultados.

As conclusões do trabalho são apresentadas no capítulo 5, logo após um breve resumo sobre o trabalho. São apresentadas também algumas propostas para trabalhos futuros.

Os anexos contém material complementar. Uma breve explicação sobre o Método dos Mínimos Quadrados e a Taxa de Redução de Erro é apresentada no Anexo I. Os alvos utilizados são mostrados no Anexo II. Por fim, o material do CD é detalhado no Anexo III.

Capítulo 2

Revisão Bibliográfica

Neste capítulo serão expostos tópicos teóricos relevantes para o entendimento do trabalho desenvolvido.

2.1 Transformação de Coordenadas

Uma forma de descrever a posição de objetos no espaço é utilizando coordenadas homogêneas. Para espaços bidimensionais, são usadas duas coordenadas com a finalidade de representar o espaço, enquanto para tridimensionais são usadas três coordenadas para tal propósito. Nesta seção, o foco será em espaços tridimensionais, considerando que o espaço bidimensional utiliza o mesmo raciocínio.

A Fig. 2.1 mostra a representação do ponto P no espaço. O ponto P está afastado da origem com distância x no eixo X, y no eixo Y e z no eixo Z. Esse ponto pode ser representado pelo vetor $(xi + yj + zk)$, sendo i, j, k vetores unitários nos eixos X, Y e Z, respectivamente.

Muitas vezes, é importante encontrar a posição de um ponto P_1 em relação a outro sistema de coordenadas, sendo necessário fazer o uso de transformações de coordenadas. Em [13], o autor mostra a teoria matemática para esse propósito. Para redefinir o sistema de referência, é preciso conhecer a translação que ocorre entre as origens dos sistemas envolvidos, assim como conhecer as novas direções dos eixos coordenados. A transformação para um novo sistema é representada por T na Eq. 2.1. A utilização de matrizes facilita a representação da transformação a partir da sequência de translações e rotações aplicadas a um sistema de coordenadas.

$$T = \left[\begin{array}{ccc|c} 3 & & & 3 \\ & \times & & \times \\ & & 3 & 1 \\ \hline 1 & \times & 3 & 1 \times 1 \end{array}\right] = \left[\begin{array}{c|c} \mathbf{Rot} & \mathbf{Trans} \\ \hline 0 & 1 \end{array}\right] \tag{2.1}$$

A translação é um deslocamento no espaço, representada pelo vetor $(p_x i + p_y j + p_z k)$, o qual quantifica o deslocamento em cada eixo coordenado. A Eq. 2.2 apresenta a transformação que representa a translação pelo vetor $(p_x i + p_y j + p_z k)$, enquanto a Fig. 2.2 mostra a representação gráfica de tal transformação.

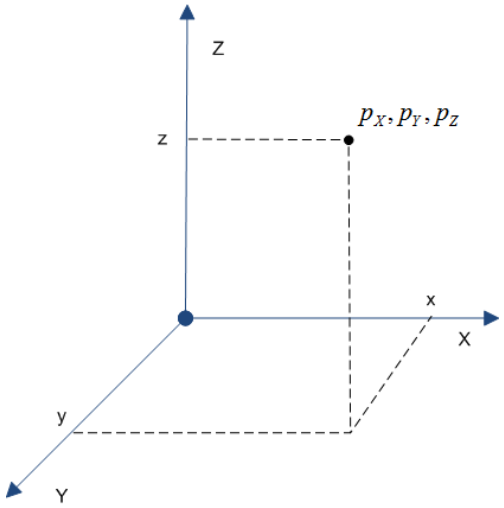


Figura 2.1: Representação de um ponto no espaço

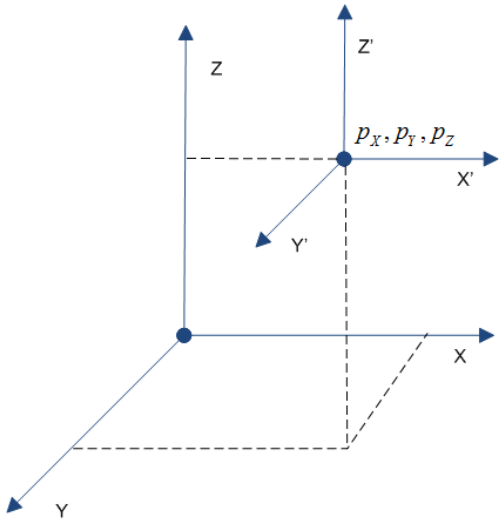


Figura 2.2: Representação de translação da origem

$$\boldsymbol{Trans}\,(p_x, p_y, p_z) = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

A Fig. 2.3 representa uma rotação do sistema de coordenadas por um ângulo de ϕ em torno do eixo Y. Transformações como essa são muito empregadas para facilitar a representação de um espaço, por exemplo.

Uma rotação pode ser representada por uma sequência de rotações em torno dos eixos coordenados, portanto, é normal ter uma representação mais complexa. É importante ressaltar que a ordem a qual ocorre a rotação sobre cada eixo influencia no resultado final. A Eq. 2.3 apresenta a transformação que rotaciona o sistema de coordenadas um ângulo ϕ em torno do eixo X. De

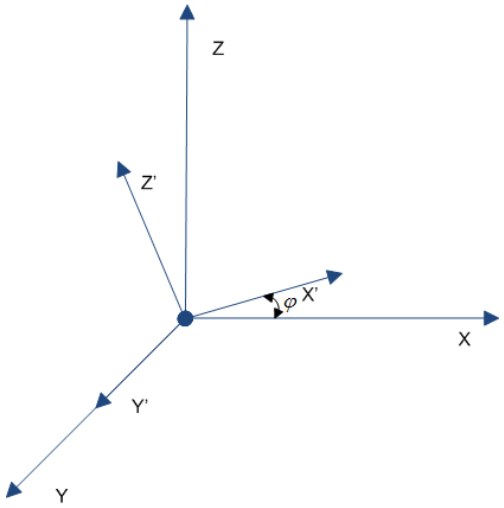


Figura 2.3: Representação de rotação em torno do eixo Y

forma similar, a Eq. 2.4 mostra a rotação de um ângulo ϕ em torno do eixo Y, enquanto a Eq. 2.5 em torno do eixo Z.

$$\boldsymbol{Rot}(X,\phi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\,(\phi) & -sen\,(\phi) & 0 \\ 0 & sen\,(\phi) & cos\,(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

$$\boldsymbol{Rot}(Y,\phi) = \begin{bmatrix} cos\,(\phi) & 0 & sen\,(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ -sen\,(\phi) & 0 & cos\,(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$\boldsymbol{Rot}(Z,\phi) = \begin{bmatrix} cos\,(\phi) & -sen\,(\phi) & 0 & 0 \\ sen\,(\phi) & cos\,(\phi) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

Para combinar uma sequência de translações e rotações, multiplicam-se as matrizes que representam cada uma das transformações envolvidas, na sequência em que ocorrem. Por exemplo, considere que a translação pelo vetor $(p_x i + p_y j + p_z k)$ seja T_1, e que a rotação do ângulo ϕ em torno do eixo Y seja R_1. Dessa forma, aplicar a transformação T_1 seguida da rotação R_1 significa aplicar a transformação resultante da multiplicação $R_1 T_1$. A Fig. 2.4 representa esse resultado. Assim, o sistema de coordenadas original é transformado para um novo sistema de coordenadas pela matriz resultante da multiplicação.

Conhecendo a transformação que muda o sistema de coordenadas N para o sistema R, e a posição de um ponto em N, é possível calcular a posição do mesmo ponto em R. A Eq. 2.6 mostra um ponto $\mathbf{q_N}$ em N (representado por $[p_{xn}\ p_{yn}\ p_{zn}\ 1]^T$) sendo transformado no ponto $\mathbf{q_R}$ em R

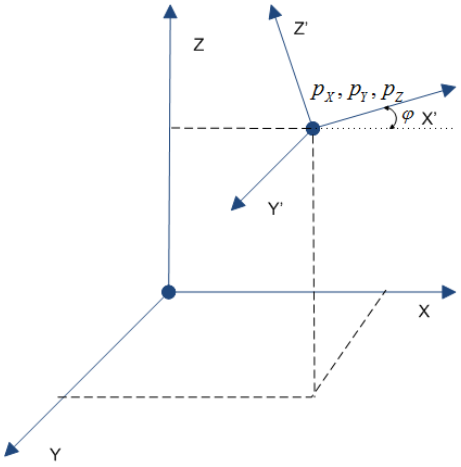


Figura 2.4: Representação de translação seguida de rotação

pela transformação T_N^R. Essa transformação não muda a posição física do ponto no espaço, muda apenas a sua posição em relação ao sistema de referência.

$$\mathbf{q_R} = T_N^R \, \mathbf{q_N} \tag{2.6}$$

A mudança do sistema de coordenadas N para o sistema de coordenadas R é apresentada na pela Eq. 2.7, e pode facilmente ser demonstrada partindo da Eq. 2.6.

$$T_R^N = \left(T_N^R\right)^{-1} \tag{2.7}$$

A transformação apresentada, relacionando sistemas de coordenadas, é chamada absoluta, e ainda é possível executá-la em sequência, passando por diversos sistemas de coordenadas. É necessário ter a precaução de multiplicar as matrizes na ordem correta, pois, por se tratar de operações matriciais, a ordem dos fatores influencia no resultado final. Por exemplo, suponha um ponto q_1, o qual passará pelas seguintes transformações (nessa ordem): T_1^2, T_2^3, T_3^4, ... T_{n-1}^n, resultando em um ponto q_n. Essa sequência de transformações é apresentada na Eq. 2.8. É possível notar que a transformação de 1 para n é representada pela Eq. 2.9.

$$q_n = T_{n-1}^n \, \ldots \, T_3^4 \, T_2^3 \, T_1^2 \, q_1 \tag{2.8}$$

$$T_1^n = T_{n-1}^n \, \ldots \, T_3^4 \, T_2^3 \, T_1^2 \tag{2.9}$$

Dessa forma, é possível encontrar um ponto em relação a outro no espaço, basta para isso redefinir a referência e aplicar a transformação necessária.

2.2 Filtro de Kalman

Para muitas aplicações, em que se modela um sistema quanto a estados, é útil conhecer o estado do sistema, seja para controle ou simplesmente para verificar se está em pleno funcionamento. Em geral, sensores são usados para essa estimação, mas há sempre um nível de incerteza associado à medida. A filtragem estocástica é utilizada para estimar o estado de um sistema a partir de observações utilizando probabilidades condicionais [14].

O Filtro de Kalman (FK) foi proposto na década de 60, e é considerado a forma ótima para estimação de estados de sistemas lineares. Neste caso, a forma ótima diz respeito à mínima variância (maior certeza). O surgimento desse algoritmo permitiu a estimação de trajetória de veículos espaciais, problema que não era resolvido até então. Assume-se que o modelo discreto do sistema pode ser representado como mostrado na Eq. 2.10 [15], sendo $\mathbf{x}$ o estado do sistema, u a sua entrada e $\mathbf{z}$ a medição realizada. Os ruídos v e w são gaussianos descorrelacionados de média nula e matriz de covariância Q e R, respectivamente. Além disso, v e w são considerados ruídos branco.

$$\begin{cases} \mathbf{x}_{k+1} = A_k\mathbf{x}_k + B_k u_k + v_k \\ \mathbf{z}_{k+1} = C_{k+1}\mathbf{x}_{k+1} + w_{k+1} \end{cases} \tag{2.10}$$

Pelo fato de haver uma incerteza associada ao estado $\mathbf{x}$, e à medida $\mathbf{z}$, eles são modelados como uma variáveis aleatórias por meio de funções de densidade de probabilidade. O uso de gaussianas para essa modelagem é comum, pois sua Função de Densidade de Probabilidade (FDP) é definida com apenas dois momentos, média e desvio padrão. Conhecendo o estado atual e as entradas, é possível estimar o próximo estado de forma a satisfazer algum critério. No caso do FK, o critério utilizado é que o estado estimado satisfaça a maior probabilidade. A Eq. 2.11 mostra a FDP de $\mathbf{x}_{k+1}$ conhecendo $\mathbf{x}_k$. Deve-se encontrar $\hat{\mathbf{x}}_{k+1}$ tal que maximize essa FDP.

$$p\left(\mathbf{x}_{k+1}|\mathbf{x}_k\right) = \frac{1}{(2\pi)^n \left|det\left(Q\right)\right|^{\frac{1}{2}}} exp\left\{-\frac{1}{2}\left(\mathbf{x}_{k+1} - A_k\mathbf{x}_k - B_k u_k\right)^T Q^{-1}\left(\mathbf{x}_{k+1} - A_k\mathbf{x}_k - B_k u_k\right)\right\} \tag{2.11}$$

Uma análise análoga é feita para a medição. A Eq. 2.12 mostra a FDP de $\mathbf{z}_{k+1}$ conhecendo $\mathbf{x}_{k+1}$, devendo obter $\hat{\mathbf{z}}_{k+1}$ que maximize essa função.

$$p\left(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}\right) = \frac{1}{(2\pi)^n \left|det\left(R\right)\right|^{\frac{1}{2}}} exp\left\{-\frac{1}{2}\left(\mathbf{z}_{k+1} - C_{k+1}\mathbf{x}_{k+1}\right)^T R^{-1}\left(\mathbf{z}_{k+1} - C_{k+1}\mathbf{x}_{k+1}\right)\right\} \tag{2.12}$$

Para exemplificar a probabilidade condicional de gaussianas, a Fig. 2.5 (adaptada de [15]) representa uma estimativa hipotética da variável aleatória x, dado as medições y_1 e y_2. O resultado de $p\left(x|y_1\right)$ tem média 0.3 e desvio padrão 0.5. A segunda medida, y_2 faz que $p\left(x|y_2\right)$ tenha média 1 e desvio padrão 0.3. Ao combinar as duas estimativas, é resultante que $p\left(x|y_1, y_2\right)$ tem média 0.74 e desvio padrão 0.19. É interessante observar que a média final ficou mais perto do resultado

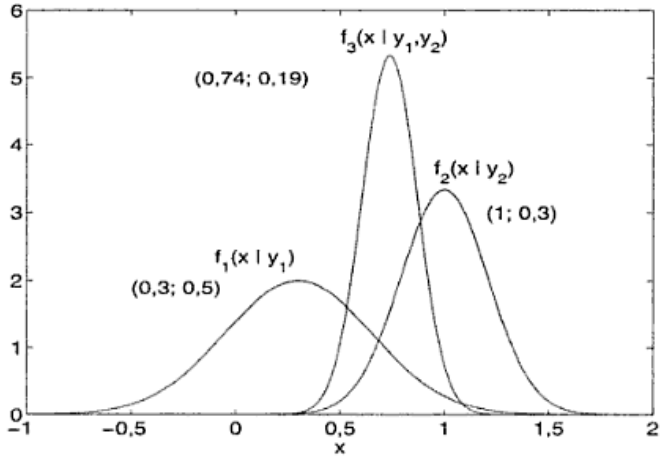


Figura 2.5: Exemplo de probabilidade condicional

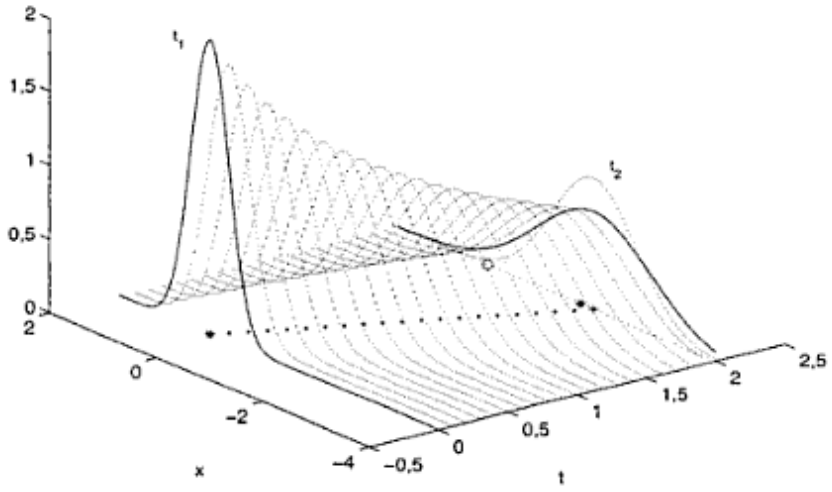


Figura 2.6: Exemplo hipotético de estimação ao longo do tempo

que se tinha maior certeza (menor desvio padrão). Outro ponto importante a ser ressaltado é que a incerteza final ficou menor do que as incertezas das medidas.

Existem duas etapas no FK, a predição e a correção. Primeiro é feita uma predição do próximo estado do sistema e de uma possível próxima medição do sensor na amostragem $k+1$. Esse estado, $\hat{\mathbf{x}}_{k+1}$, é tal que maximiza a FDP apresentada na Eq. 2.11. Quando há medição, ela age corrigindo a predição feita, tendendo a diminuir a incerteza daquele estado. A Fig. 2.6 (adaptada de [15]) mostra a estimação do estado ao longo do tempo. Em t_1 é realizada uma medida, gerando um alto nível de certeza sobre o estado. Ao longo do tempo, não são feitas medidas, aumentando o desvio padrão da gaussiana, ou seja, a incerteza sobre o estado atual. Em t_2, é realizada uma nova medida, permitindo que a correção seja feita, e, consequentemente, aumentando a certeza sobre o estado.

Para estimar $\mathbf{x}_{k+1}$, o próximo estado, aplica-se o estado atual, $\mathbf{x}_k$, e as entradas na primeira parte da Eq. 2.10. A esperança de v_{k+1} é nula. Dessa forma, a propagação é feita como mostrado

na Eq. 2.13.

$$\hat{\mathbf{x}}_{k+1|k} = A_k \mathbf{x}_k + B_k u_k \tag{2.13}$$

A matriz de covariância do próximo estado é apresentada na Eq. 2.14. É possível perceber que, na fase de predição, a incerteza tende a aumentar em relação ao estado passado.

$$P_{k+1|k} = A_k P_k A_k^T + Q \tag{2.14}$$

A etapa de correção é feita utilizando a diferença entre o valor da medição e o valor esperado para a medição dado $\hat{x}_{k+1|k}$. Essa diferença é chamada inovação, e ela deve ser multipliaca pelo termo denominado Ganho de Kalman. A etapa de corrreção é apresentada na Eq. 2.15.

$$\hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_{k+1|k} + K_{k+1}\left[\mathbf{z}_{k+1} - C_{k+1}\hat{\mathbf{x}}_{k+1|k}\right] \tag{2.15}$$

Quando é feita uma correção, a incerteza que se tinha diminui. A atualização do valor da incerteza é apresentado na Eq. 2.16.

$$P_{k+1} = \left(I - K_{k+1}C_{k+1}\right) P_{k+1|k} \left(I - K_{k+1}C_{k+1}\right)^T + K_{k+1} R K_{k+1}^T \tag{2.16}$$

A cada iteração, deve-se definir o Ganho de Kalman. Para garantir que o FK seja ótimo, K_{k+1} deve ser tal que minimize a variância da estimativa, ou seja, minimize P_{k+1}. A Eq. 2.17 mostra a derivada de P_k em relação a K_k.

$$\frac{\partial P}{\partial K_{k+1}} = -2\left(I - K_{k+1}C_{k+1}\right) P_{k+1|k} C_{k+1}^T + 2K_{k+1}R \tag{2.17}$$

Ao igualar a Eq. 2.17 a zero, é possível encontrar K_{k+1} que minimize P_{k+1}, mostrado na Eq. 2.18.

$$K_{k+1} = P_{k+1|k} C_{k+1}^T \left[C_{k+1}\hat{P}_{k+1}C_{k+1}^T + R\right]^{-1} \tag{2.18}$$

Utilizando a Eq. 2.18, a Eq. 2.16 pode ser reescrita como apresentada na Eq. 2.19.

$$P_{k+1} = P_{k+1|k} - K_{k+1}C_{k+1}P_{k+1|k}^T \tag{2.19}$$

Em resumo, o algoritmo do FK é apresentado na Eq. 2.20.

$$\begin{cases} \hat{\mathbf{x}}_{k+1|k} = A_k \mathbf{x}_k + B_k u_k \\ P_{k+1|k} = A_k P_k A_k^T + Q \\ K_{k+1} = P_{k+1|k} C_{k+1}^T \left[C_{k+1} P_{k+1|k} C_{k+1} T + R\right]^{-1} \\ \hat{\mathbf{x}}_{k+1} = \hat{\mathbf{x}}_{k+1|k} + K_{k+1}\left[\mathbf{z}_{k+1} - C_{k+1}\hat{x}_{k+1|k}\right] \\ P_{k+1} = P_{k+1|k} - K_{k+1}C_{k+1}P_{k+1|k} \end{cases} \tag{2.20}$$

O FK é um algoritmo muito difundido, porém, ele é utilizado apenas para sistemas lineares. Para estimar estados de sistemas não lineares, é necessário utilizar outras abordagens, como aproximações analíticas ou métodos de Monte Carlo, o que pode ser inviável em diversas aplicações devido ao esforço computacional. O modelo discreto do sistema não linear é apresentado na Eq. 2.21, e é similar ao modelo mostrado na Eq. 2.10, porém, f e h podem ser funções não lineares. Os ruídos v_k e w_k são descorrelacionados e têm média nula e matriz de covariância Q e R, respectivamente. A seguir são apresentadas duas abordagens para sistemas não lineares, o Filtro de Kalman Estendido (FKE) e o Filtro de Kalman *Unscented* (FKU).

$$\begin{cases} \mathbf{x}_{k+1} = f\left(\mathbf{x}_k, u_{k+1}, v_{k+1}\right) \\ z_{k+1} = h\left(\mathbf{x}_{k+1}, w_{k+1}\right) \end{cases} \tag{2.21}$$

2.2.1 Filtro de Kalman Estendido

Um dos métodos de filtragem mais utilizados para sistemas não lineares é o Filtro de Kalman Estendido (FKE). O FKE lineariza o sistema em torno do ponto de operação, permitindo o uso do algoritmo do FK. A linearização é feita utilizando as jacobianas de f e h, linearizando o problema. A matriz jacobiana da função f, $Df\left(x\right)$ é apresentada na Eq. 2.22.

$$Df\left(\mathbf{x}\right) = \begin{bmatrix} \frac{\partial f_1}{\partial \mathbf{x}_1} & \frac{\partial f_1}{\partial \mathbf{x}_2} & \cdots & \frac{\partial f_1}{\partial \mathbf{x}_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial \mathbf{x}_1} & \frac{\partial f_n}{\partial \mathbf{x}_2} & \cdots & \frac{\partial f_n}{\partial \mathbf{x}_n} \end{bmatrix} \tag{2.22}$$

De forma análoga, pode-se calcular a jacobiana com respeito ao ruído do processo. A Eq. 2.23 apresenta $Df\left(w\right)$.

$$Df\left(v\right) = \begin{bmatrix} \frac{\partial f_1}{\partial v_1} & \frac{\partial f_1}{\partial v_2} & \cdots & \frac{\partial f_1}{\partial v_n} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial f_n}{\partial v_1} & \frac{\partial f_n}{\partial v_2} & \cdots & \frac{\partial f_n}{\partial v_n} \end{bmatrix} \tag{2.23}$$

A matriz jacobiana de h, representada por $Dh\left(\mathbf{x}\right)$, é calculada de forma análoga. No FKE, as jacobianas apresentadas estão em função de x, e devem ser calculadas a cada iteração. Ao utilizar $x = x_k$ nas matrizes jacobianas apresentadas, as funções f e h são linearizadas em torno de x_k.

A Eq. 2.24 mostra o algoritmo para o FKE.

$$\begin{cases} \hat{\mathbf{x}}_{k+1|k} = f\left(\mathbf{x}_k, u_k\right) \\ P_{k+1|k} = Df\left(\mathbf{x}_k\right) P_k Df\left(\mathbf{x}_k\right)^T + Df\left(w_k\right) Q Df\left(w_k\right)^T \\ K_{k+1} = P_{k+1} Dh\left(\hat{\mathbf{x}}_{k+1|k}\right)^T \left[Dh\left(\hat{\mathbf{x}}_{k+1|k}\right) Dh\left(\hat{\mathbf{x}}_{k+1|k}\right)^T + Dh\left(\hat{\mathbf{x}}_{k+1|k}\right) R Dh\left(\hat{\mathbf{x}}_{k+1|k}\right)^T\right]^{-1} \\ \mathbf{x}_{k+1} = \hat{\mathbf{x}}_{k+1} + K_{k+1}\left[\mathbf{z}_{k+1} - Dh\left(\hat{\mathbf{x}}_{k+1}\right)\right] \\ P_{k+1} = \hat{P}_{k+1} - K_{k+1} Dh\left(\hat{\mathbf{x}}_{k+1}\right) \hat{P}_{k+1} \end{cases} \tag{2.24}$$

2.2.2 Filtro de Kalman *Unscented*

Segundo a referência [16], apesar de muito utilizado, o FKE apresenta problemas bem conhecidos:

- A linearização pode deixar o filtro instável, caso o intervalo de amostragem não seja suficientemente pequeno.
- O cálculo da matriz jacobiana do sistema não é prático, inclusive podendo não ser definida no ponto de operação.

Um método geralmente mais eficiente computacionalmente do que o FKE é o Filtro de Kalman *Unscented* (FKU), o qual utiliza a Transformação *Unscented* (TU), uma forma determinística para escolha de amostras que representam os primeiros momentos da distribuição. Essa abordagem é similar ao filtro de partículas, o qual também utiliza a amostragem, porém, com a TU, as amostras são definidas de forma determinística.

2.2.2.1 Transformação *Unscented*

Em [17], o autor define a TU. Partindo do princípio de que com um número fixo de parâmetros é mais simples aproximar uma distribuição gaussiana do que uma função não linear, é possível obter um conjunto de dados que mantenha os primeiros momentos de uma distribuição original, e possa ser propagado por uma transformação não linear. Dessa forma, um conjunto de variáveis de estado é aproximado por um conjunto de amostras determinísticas χ denominadas pontos sigma [18]. Quando esses pontos são propagados por uma função não linear, a média e a variância do resultado representam melhor a variável original propagada por essa transformação. Em [1], para mostrar a diferença entre os resultados das duas técnicas possíveis, o autor fez uma simulação transformando pontos de uma amostragem por uma função g não linear, fazendo uma comparação entre a linearização e a TU, o resultado dessa simulação é apresentado na Fig. 2.7. É possível perceber que a linearização tem um erro associado maior do que a TU.

Para um conjunto de n variáveis de estado representado por uma matriz de dimensão $n \times 1$, com média $\bar{x}$ e matriz de covariância P, de dimensão $n \times n$, é necessário um conjunto χ_k^* com $2n + 1$ pontos, que são gerados como mostrados na Eq. 2.25, onde κ é um fator de ponderação que influencia na distribuição dos pontos em torno da média. É usual que o valor de κ seja tal que $n + \kappa = 3$, mas existem outras abordagens para a definição dessa ponderação. Para computar a raiz quadrada matricial, um método eficiente é a Decomposição de Cholesky.

$$\chi_k^* = \left(\mathbf{x}_k;\, \mathbf{x}_k + \sqrt{(n+\kappa)\,P_k};\, \mathbf{x}_k - \sqrt{(n+\kappa)\,P_k}\right) \tag{2.25}$$

Cada ponto sigma gerado deve ser transformado pela função f, não linear, como mostra a Eq. 2.26.

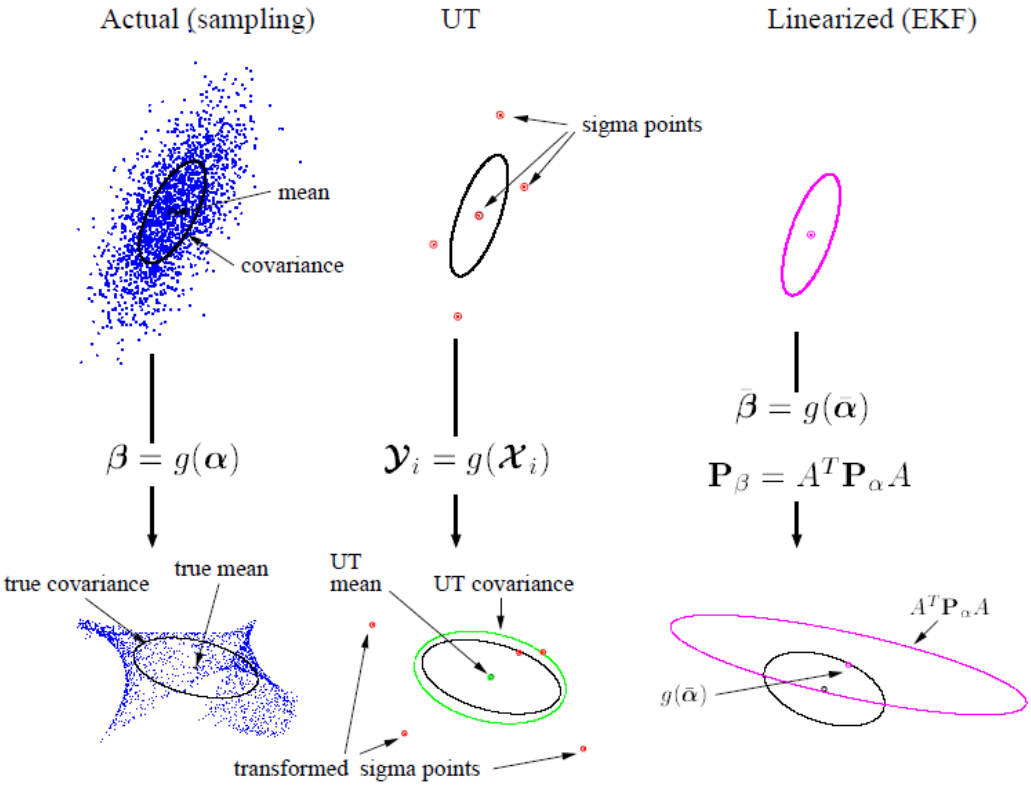


Figura 2.7: Comparação entre linearização e TU - Imagem retirada de [1]

$$\chi_{k+1}^{[i]} = f\left(\chi_k^{[i]}, u_k\right), \; 0 \leq i \leq 2n \tag{2.26}$$

A média da estimativa é feita como mostra a Eq. 2.27, enquanto o cálculo da covariância é mostrado na Eq. 2.28.

$$\bar{\chi}_k = \frac{1}{n+\kappa}\left\{\kappa\,\chi_k^0 + \frac{1}{2}\sum_{i=1}^{2n}\chi_k^i\right\} \tag{2.27}$$

$$P_{xx} = \frac{1}{n+\kappa}\left\{\kappa\left[\chi_k^0 - \bar{\chi}_k\right]\left[\chi_k^0 - \bar{\chi}_k\right]^T + \frac{1}{2}\sum_{i=1}^{2n}\left[\chi_k^i - \bar{\chi}_k\right]\left[\chi_k^i - \bar{\chi}_k\right]^T\right\} \tag{2.28}$$

Ao se utilizar a linearização por expansão em Série de Taylor em torno da média $\bar{x}$, é introduzido um erro na aproximação, enquanto com a TU, esse erro de aproximação é menor. Outra vantagem do FKU, além de não necessitar calcular a jacobiana do sistema, é a possibilidade de uso mesmo quando o ponto de operação é uma discontinuidade, o que causaria problemas se o método de linearização estivesse sendo empregado, pois a jacobiana não seria definida em tal ponto, e assim não podendo prever a covariância.

2.2.2.2 O Algoritmo - FKU

Em [19], o autor define o algoritmo para o FKU. O problema consiste em utilizar um filtro recursivo para o sistema representado pela Eq. 2.21. Em [20] o autor mostra o conjunto de equações recursivas simples e didático para a utilização de tal filtro.

1. São computados os pontos sigma do vetor de estado no instante k, gerando o conjunto χ_k^*, como mostrado na Eq. 2.29.
2. Cada um dos pontos resultantes χ_k^i deverá ser transformado pela função f, junto com as entradas u_k obtidas no instante k, resultando no conjunto $\left\{\hat{\chi}_{k+1|k}^*\right\}$, como mostrado na Eq. 2.30.
3. Estima-se a média do estado, $\hat{\mathbf{x}}_{k+1|k}$, como mostrado na Eq. 2.31, sendo w_m uma ponderação tal que $w_m = \kappa/n + \kappa$ para $i = 0$, e $w_m = \left(1 - w_m^0\right)/2n$.
4. Calcula-se a matriz de covariância do estado, $P_{k+1|k}$, como mostrado na Eq. 2.32.
5. A partir de $\hat{\mathbf{x}}_{k+1|k}$ e de $P_{k+1|k}$, é calculado um novo conjunto de pontos sigma, $\left\{\hat{\chi}_{k+1}^*\right\}$, de acordo com a Eq. 2.33.
6. Os pontos do conjunto $(\hat{\chi}_k^*)$ são transformados pela função h, gerando o conjunto $\hat{\mathbf{z}}_{k+1}$, como mostrado na Eq. 2.34.
7. A estimativa para a medida, $\hat{\mathbf{z}}_{k+1}$, é calculada, também por uma ponderação dos pontos sigma, como mostrado na Eq. 2.35.
8. Calcula-se a covariância da medida, S_{k+1}, de acordo com a Eq. 2.36.
9. É calculada a covariância cruzada, P_{xz}, de acordo com a Eq. 2.37.
10. É calculado o ganho de Kalman, K_k, como mostrado na Eq. 2.38.
11. A estimativa do estado é corrigida, determinando $\hat{\mathbf{x}}_{k+1}$, de acordo com a Eq. 2.39.
12. A matriz de covariância P_{k+1} é ajustada como na Eq. 2.40.

$$\chi_k^* = \left(\mathbf{x}_k;\ \mathbf{x}_k + \sqrt{(n+\kappa)\,P_k};\ \mathbf{x}_k - \sqrt{(n+\kappa)\,P_k}\right) \tag{2.29}$$

$$\hat{\chi}_{k+1|k}^* = f\left(u_k, \chi_k^*\right) \tag{2.30}$$

$$\hat{\mathbf{x}}_{k+1|k} = \sum_{i=0}^{2n} w_m^{[i]} \hat{\chi}_{k+1|k}^{[i]} \tag{2.31}$$

$$P_{k+1|k} = \sum_{i=0}^{2n} w_m^{[i]} \left(\hat{\chi}_{k+1|k}^{*[i]} - \hat{\mathbf{x}}_{k+1|k}\right) \left(\hat{\chi}_{k+1|k}^{*[i]} - \hat{\mathbf{x}}_{k+1|k}\right)^T + Q \tag{2.32}$$

$$\hat{\chi}_{k+1}^* = \left(\hat{\mathbf{x}}_{k+1|k};\ \hat{\mathbf{x}}_{k+1|k} + \sqrt{(n+\kappa)\,P_{k+1|k}};\ \hat{\mathbf{x}}_{k+1|k} - \sqrt{(n+\kappa)\,P_{k+1|k}}\right) \tag{2.33}$$

$$\hat{\mathbf{z}}^*_{k+1} = h\left(\bar{\chi}^*_{k+1}\right) \tag{2.34}$$

$$\hat{\mathbf{z}}_{k+1} = \sum_{i=0}^{2n} w_m^{[i]} \bar{\mathbf{z}}^{[i]}_{k+1} \tag{2.35}$$

$$S_{k+1} = \sum_{i=0}^{2n} w_m^{[i]} \left(\bar{\mathbf{z}}^{[i]}_{k+1} - \hat{\mathbf{z}}_{k+1}\right)\left(\bar{\mathbf{z}}^{[i]}_{k} - \hat{\mathbf{z}}_{k}\right)^T + R \tag{2.36}$$

$$P_{xz} = \sum_{i=0}^{2n} w_m^{[i]} \left(\bar{\chi}^{[i]}_{k} - \bar{\mathbf{x}}_{k}\right)\left(\bar{\mathbf{z}}^{[i]}_{k} - \hat{\mathbf{z}}_{k}\right)^T \tag{2.37}$$

$$K_{k+1} = P_{xz} S_k^{-1} \tag{2.38}$$

$$\mathbf{x}_{k+1} = \bar{\mathbf{x}}_k + K_k\left(\mathbf{z}_k - \hat{\mathbf{z}}_k\right) \tag{2.39}$$

$$P_{k+1} = \bar{P}_k - K_k S_k K_k^T \tag{2.40}$$

É importante ressaltar o ajuste correto das matrizes de covariância. Por se tratar de uma análise probabilística, o conhecimento sobre a incerteza associada a cada dado é muito importante.

2.2.3 Adequação do Filtro

O ajuste das matrizes de covariância é crucial para o bom funcionamento do filtro. Seu significado físico representa o quanto se confia no processo e na medição.

A matriz R, covariância do ruído de medição, pode ser modelada como a incerteza do sensor. Uma boa escolha é usar a variância das medidas para modelar R. O ajuste da matriz Q, covariância do ruído do processo, possui uma modelagem mais complexa. Uma boa abordagem se faz analisando a inovação e a incerteza sobre a medida. Aproximadamente 95% dos pontos devem ficar dentro do intervalo $\pm 3\sigma$. Se mais pontos ficarem dentro do intervalo, significa que a matriz Q está sendo sobre estimada, enquanto o contrário significa uma sub estimação.

Outro ajuste interessante é saber quando aceitar uma medida. Esse ajuste pode ser feito como mostra a Eq. 2.41, sendo S a variância da medida. A inovação, definida como a subtração $z_{k+1} - \hat{z}_{k+1}$, deve satisfazer tal condição para que ela seja aceita. A constante l depende da dimensão de z_{k+1}

$$\left[z_{k+1} - \hat{z}_{k+1}\right]^T S^{-1} \left[z_{k+1} - \hat{z}_{k+1}\right] \leq l \tag{2.41}$$

2.2.4 Estimação de Parâmetros

Filtros recursivos podem ser utilizados para estimar os parâmetros da função em conjunto com os estados [15]. Para essa consideração, é necessário que a variação dos parâmetros seja nula ou muito mais lenta do que a variação dos estados. A representação em espaço de estados para os parâmetros do sistema são mostrados na Eq. 2.42, sendo p_k o parâmetro estimado no instante k, mantendo o valor que tinha no instante $k-1$ perturbado por um ruído r_k, de média nula e matriz

de covariância r. Ao realizar a estimação de estado e de parâmetros, o vetor de estados deve ser estendido, passando a conter os parâmetros. A equação de medição depende do estado atual, $\mathbf{x}_k$ e do vetor de estados do parâmetros estimado p_k, ambos no instante k. Para sistemas invariantes no tempo, essa abordagem permite que erros de modelo sejam ajustados junto com a estimativa do vetor de estados. Em sistemas variantes, é possível acompanhar a mudança de seus parâmetros, desde que ela evolua muito mais lentamente do que o vetor de estados.

$$\begin{cases} p_k = p_{k-1} + r_k \\ \mathbf{z}_k = G\left(\mathbf{x}_k, w_k\right) + e_k \end{cases} \tag{2.42}$$

2.3 Controle de Trajetória de Veículos Auto Guiados

O controle de trajetória é muito estudado em robótica móvel. Tendo o ponto de partida de um robô e seu ponto final desejado, existem diversos caminhos possíveis. O percurso a ser utilizado é concebido na fase de navegação. A navegação pode levar em conta alguns fatores externos, como obstáculos no caminho ou menor distância a percorrer, por exemplo. A teoria desta seção foi apresentada na referência [21].

Definida a trajetória, é necessário manter o robô em seu caminho, sendo utilizados sensores e atuadores para tal finalidade. Os sensores permitem que seja verificada a posição corrente do robô, analisando seu desvio da trajetória, enquanto os atuadores agem para que seja feita a devida correção. É muito comum a utilização de técnicas de controle não linear para o controle de trajetória, como por exemplo redes neurais artificiais e lógica nebulosa, viabilizando a estabilidade do sistema.

Tendo a medição da postura do robô (posição e orientação), é possível calcular o quanto ele está afastado da trajetória desejada. O controlador atua sobre o robô, reduzindo o erro no seu percurso. O desvio do caminho pode ser representado pela distância entre o robô e o traço da trajetória, e pelo seu desvio angular entre a orientação do robô e a tangente do percurso. A Fig. 2.8 mostra um robô com tração diferencial, seguindo com velocidade v, na posição (x', y'), ele está a uma distância Γ de sua trajetória, e um desvio angular Θ.

Um diagrama do funcionamento padrão do controle de trajetória pode ser visualizado na Fig. 2.9. Em um nível mais alto, a trajetória é definida pelo bloco de navegação. Com a definição do caminho, são utilizados sensores para calcular o desvio do percurso original. A informação sobre o erro e sobre a trajetória são passados para um nível intermediário, que gera um sinal de referência para a velocidade das rodas. Em um nível mais baixo é feita a atuação nas rodas. A partir da referência gerada, é utilizado um controlador de velocidade para cada roda, permitindo assim que o robô siga sua trajetória. Esta seção foca no bloco intermediário, o qual gera a atuação dados o erro e a trajetória.

Um robô com tração diferencial tem raio r para as rodas e b de distância entre elas. Sua postura é definida por (x, y, θ), sua cinemática depende da velocidade de rotação das rodas da seguinte maneira:

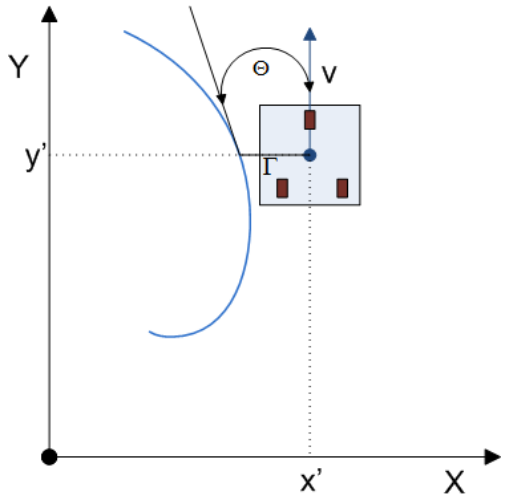


Figura 2.8: Exemplo de robô fora da trajetória

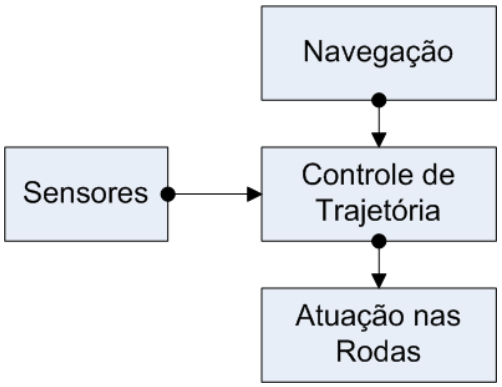


Figura 2.9: Diagrama controlador de trajetória

$$\dot{x}(t) = v(t)\, cos(\theta(t)) \tag{2.43}$$

$$\dot{y}(t) = v(t)\, sen(\theta(t)) \tag{2.44}$$

$$\dot{\theta}(t) = \frac{\omega_r(t) - \omega_l(t)}{b} r \tag{2.45}$$

$$v(t) = \frac{\omega_r(t) + \omega_l(t)}{2} r \tag{2.46}$$

Sendo v a velocidade linear do robô, $\dot{x}$ a velocidade do robô no eixo X, $\dot{y}$ a velocidade do robô no eixo Y, $\dot{\theta}$ sua velocidade angular, e ω_r e ω_l as velocidades angulares das rodas direita e esquerda, respectivamente.

As velocidades angulares geradas pelo controlador de trajetória são apresentadas na Eq. 2.47, sendo r o raio da roda e v a velocidade de navegação do robô. A função $\Delta\omega(\Gamma(t), \Theta(t))$, que é dependente do desvio da trajetória, permite que o robô se curve para adequação à trajetória.

$$\left\{ \begin{array}{l} \omega_r = v/r + \Delta\omega(\Gamma(t), \theta(t)) \\ \omega_l = v/r - \Delta\omega(\Gamma(t), \theta(t)) \end{array} \right\} \tag{2.47}$$

É utilizado um controlador proporcional para que seja feito o controle da trajetória. Dessa forma, a lei de controle é definida como mostra a Eq. 2.48.

$$\Delta\omega(\Gamma(t), \Theta(t)) = K_\Gamma \Gamma(t) + K_\Theta \Theta(t) \tag{2.48}$$

Na referência [21], os autores fazem a análise de estabilidade do sistema, desenvolvê-la aqui fugiria do escopo do trabalho. Para que o acompanhamento da trajetória ocorra rapidamente e sem sobrepasso, é definida a Eq. 2.49 como relação entre os ganhos e as dimensões do robô. Essa relação depende apenas de parâmetros do robô, não variando no tempo.

$$K_\Theta = \sqrt{\frac{2 \cdot v \cdot K_\Gamma \cdot b}{r}} \tag{2.49}$$

Capítulo 3

Sistema de Localização Auxiliar Com Realidade Aumentada

Neste capítulo serão apresentados aspectos práticos relativos ao desenvolvimento do sistema de localização auxiliar (com realidade aumentada) proposto para o trabalho em torno de localização com RFID, e o controle de trajetórias desenvolvido no robô móvel para testes. Aqui serão apresentadas as ferramentas utilizadas, e serão mostrados os experimentos desenvolvidos e seus resultados.

3.1 Introdução

Com o objetivo de localizar o robô móvel, foram desenvolvidos dois diferentes sistemas de localização. O primeiro utiliza visão computacional, e tem resultados excelentes, podendo ser utilizado para testar ou combinar-se ao segundo, o qual utiliza RFID. O trabalho foi realizado no Laboratório de Automação e Robótica da UnB. O LARA tem dimensão total de 11 m $\times$ 18 m, porém, apenas uma fração foi utilizada para os testes. A Fig. 3.1 mostra o laboratório e as convenções de orientação adotadas, destacando em vermelho a região utilizada para os testes.

Em alguns diagramas ao longo do texto, para melhor entendimento, uma trajetória pode aparecer ilustrada sem estar sobreposta ao mapa do laboratório como na Fig. 3.1. Mesmo nestes casos, o sistema de coordenadas (que pode aparecer como na forma clássica: com a origem no canto inferior esquerdo da imagem) corresponde ao sistema de coordenadas absoluto definido e ilustrado na Fig. 3.1.

3.2 Robô Móvel

Para o desenvolvimento do trabalho, foi utilizado o robô *Aramis* [22], plataforma didática de robótica da marca *Pioneer*. Suas dimensões são explicitadas na Fig. 3.2 (adaptada do manual

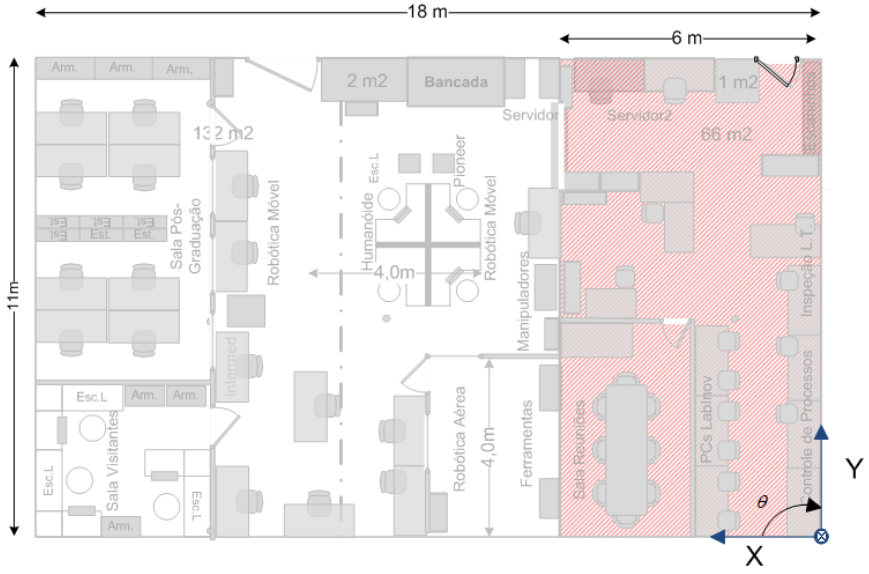


Figura 3.1: Representação do LARA

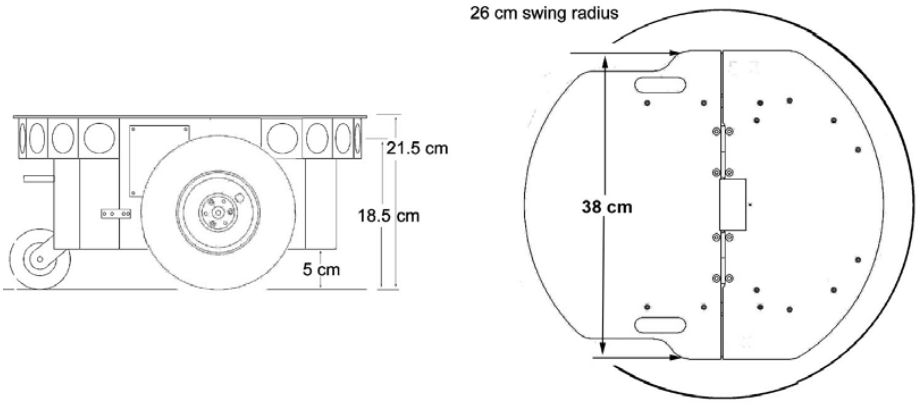


Figura 3.2: Dimensões do robô móvel *Aramis*

[22]). O *Aramis* possui duas rodas diferenciais e uma roda *caster*, com dois graus de liberdade. Uma representação da roda do tipo *caster* pode ser visualizada na Fig. 3.3. O robô tem um computador embarcado com sistema operacional Linux em tempo real, e se comporta como um servidor para comunicação com outros computadores. A comunicação é feita por TCP/IP, e faz uso de uma antena *wi-fi* presente no *Aramis*. A plataforma é dotada de sensores, como sonar e *encoders*, que recebem comando de um microprocessador, o qual utiliza comunicação serial para trocar informações com o computador.

Utilizando os dados sensoriais dos *encoders*, é possível determinar a pose do robô ao longo do tempo. O modelo discreto para a evolução do sistema é mostrado na Eq. 3.1, sendo l a distância do centro do robô para as rodas, dt o período de amostragem dos *encoders*, e, v_r e v_l as velocidades de rotação das rodas direita e esquerda, respectivamente. As variáveis de estado (x_k, y_k, θ_k) são as coordenadas em x e y e a orientação do robô no laboratório, respectivamente, conforme as direções convencionadas na Fig. 3.1. É importante ressaltar que, a partir dos dados dos *encoders*,

a interface de captura de dados implantada na plataforma em trabalhos anteriores, retorna as velocidades lineares das rodas, não as angulares.

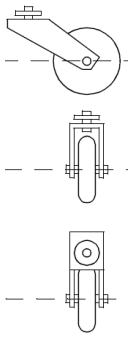


Figura 3.3: Representação da roda tipo *caster*

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} cos\,(\theta_{k-1}) & -sen\,(\theta_{k-1}) & 0 \\ sen\,(\theta_{k-1}) & cos\,(\theta_{k-1}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{v_r+v_l}{2} \\ 0 \\ \frac{-v_r+v_l}{2l} \end{bmatrix} \cdot dt \tag{3.1}$$

No manual do robô [22], o fabricante alerta sobre a utilização da integração das velocidades para obter a posição. Devido a fatores como escorregamento, esse sistema é confiável apenas para pequenas distâncias. Ao haver um erro no posicionamento, esse erro será integrado ao longo do tempo, acumulando o erro de posição.

Foi realizado um teste inicial para verificar a necessidade da utilização de sensores adicionais no robô. Partindo de uma posição inicial conhecida, aproximadamente $(1550, 500)$, em milímetros, o robô fez uma trajetória durante aproximadamente 665 s, controlado remotamente. Utilizando apenas a Eq. 3.1 com os dados obtidos dos *encoders*, o resultado de localização utilizando apenas o recurso de odometria mostra que o robô passa pela posição $(3393.0, -95.8)$, como mostrado na Fig. 3.4. Esse resultado é absurdo, pois coordenadas negativas remetem ao lado externo do laboratório.

Observando o resultado obtido com a odometria do robô sem correção de sensores, é importante que seja definido um resultado mais confiável para comparação e validação da localização. Para o sistema confiável, foi desenvolvido um sistema de localização utilizando visão computacional em conjunto com uma ferramenta de realidade aumentada, o *ARToolKit*.

3.3 *ARToolKit*

Dentro do contexto de localização para robótica móvel, foi proposto um sistema de localização para a plataforma *Aramis* com base na tecnologia de realidade aumentada. Esse sistema foi escolhido por se esperar um resultado com alto grau de precisão nas medidas. O resultado da implementação é utilizado para comparação (*ground truth*) ou combinado ao sistema de localização com sinais de potência de RFID, fornecendo assim resultados ainda melhores.

Realidade Aumentada (AR - *Augmented Reality*) é um termo utilizado para a percepção do

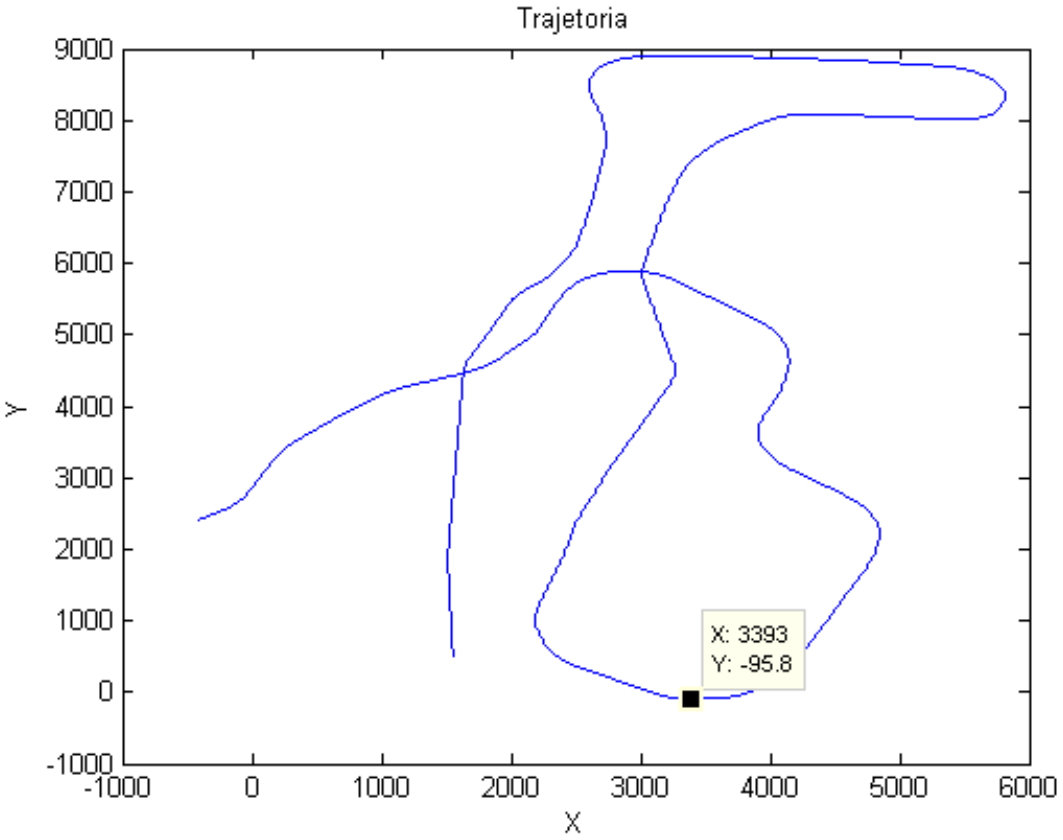


Figura 3.4: Localização do robô numa trajetória utilizando apenas odometria.

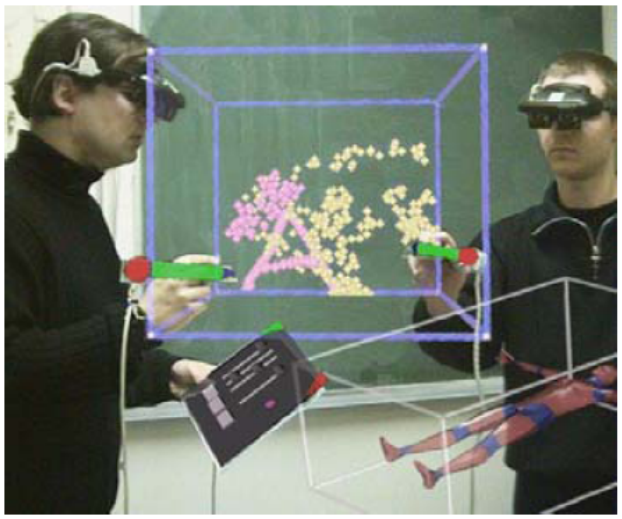


Figura 3.5: Exemplo de aplicação com realidade aumentada por Visão. Com o auxílio de displays montados sobre os olhos, objetos virtuais são posicionados na cena.

mundo real modificado por elementos sensitíveis gerados por computador, como som, vídeo e até mesmo odores. Fazendo um paralelo com o conhecido conceito de Realidade Virtual, que tem como premissa a criação de um mundo completamente artificial, a Realidade Aumentada visa mesclar sensações reais do mundo a sensações virtuais geradas digitalmente. Para tanto, sistemas compostos de funcionalidades de realidade aumentada precisam encontrar uma ligação entre o fator real e o fator virtual a serem combinados. Por exemplo, no caso de realidade aumentada especificamente por visão (nosso objeto de interesse), é preciso ligar os elementos gráficos gerados artificialmente por computador à mesma perspectiva de visão do agente que observa as imagens reais (câmeras, HMD's - *head mounted displays*, o olho humano, e outros). O "aumento" da percepção (realidade) é convencionalmente em tempo real e em contexto semântico com os elementos ambientais, como marcas pré-definidas, elementos gráficos transmitidos por televisão, faces em rastreamento para segurança, entre muitas outras aplicabilidades. A Fig. 3.5 (adaptada da tese de doutorado [23]) apresenta um exemplo de utilização de realidade aumentada.

O interesse no sistema de realidade aumentada para construção de um mecanismo de localização para robô móvel está associado a essa busca contínua (em tempo real) do elo de ligação entre mundo externo e mundo digital. No caso, este relacionamento se dá na posição relativa de marcas pré-definidas, presentes numa imagem captada, com respeito à câmera que gera essas mesmas imagens. Essa informação relativa, em conjunto à informação da posição absoluta de cada marca no sistema de coordenadas do ambiente (intrínseco à montagem do sistema), fornece uma estimativa da posição absoluta da câmera. Por conseguinte, obtém-se a posição do robô, pois sabe-se sua posição com respeito ao sistema de coordenadas da câmera ou qualquer ponto descrito no sistema de coordenadas da câmera. A fase de estimação da posição relativa entre câmera e marcação, parte constituinte do algoritmo de realidade aumentada, funcionará como um sensor de posição para o sistema de localização.

ARToolKit (*Augmented Reality Toolkit*), ou kit de ferramentas para realidade aumentada, é

uma biblioteca de software em *C/C++* que é disponibilizado livremente para uso não-comercial sob a licença para publico geral (*GNU General Public License*). Licenças comerciais para uma implementação profissional do *ARToolKit* são disponíveis a usuários para os quais não se aplica a licença GPL, ou que queiram um suporte de alto nível (licenças comerciais são administradas pela *ARtoolworks*, Inc., Seatle, WA, USA.). Essa biblioteca de software permite ao usuário o desenvolvimento de aplicações que façam uso de realidade aumentada de forma simples. Uma das partes mais importantes e críticas do desenvolvimento de realidade aumentada por visão é determinar precisamente e a cada instante a posição e orientação do usuário com respeito a marcações na imagem, de forma que seja possível posicionar objetos virtuais na posição exata dentro de um quadro, para que sua projeção esteja alinhada com os objetos do mundo real captados na imagem. O *ARToolKit* encapsula técnicas e funcionalidades de visão computacional que determinam a posição e orientação da câmera com respeito a marcas predefinidas (semelhantes à marca na Fig. 3.6), permitindo ao usuário sobrepor imagens geradas artificialmente a essas marcas.

Figura 3.6: Exemplo de alvo de rastreamento para uso com *ARToolKit*

Com base nas informações colocadas, é direta a justificativa da proposição de um sistema de localização utilizando realidade aumentada. Uma vez que a ferramenta de software (*ARToolKit*) estima a pose relativa da câmera a diferentes marcas nas imagens, dados alvos de rastreamento fixos com posição conhecida, é possível também inferir a posição e orientação do robô (*Aramis*) no ambiente (LARA). Contudo, o desenvolvimento desse sistema de localização não se deu pela simples utilização da interface das bibliotecas do *ARToolKit*. Como será detalhado adiante, uma série decisões e considerações quanto à montagem e ao desenvolvimento desse sistema foram levadas em conta com base em aspectos mais profundos do funcionamento do *ARToolKit*.

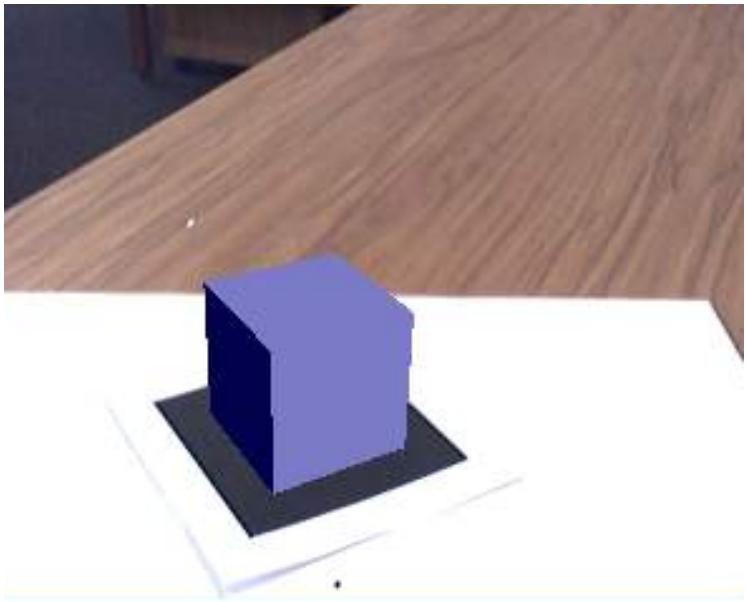


Figura 3.7: Exemplo de aplicação simples com sobreposição de imagem virtual sobre imagem real utilizando o *ARToolKit*

3.3.1 Instalação na Plataforma *Aramis*

O primeiro passo do desenvolvimento do sistema de localização por realidade aumentada é a instalação do software utilizado (*ARToolKit*) na plataforma *Aramis*. Em essência, o *ARToolKit* é uma coleção de bibliotecas de software, projetadas para serem ligadas a programas usuários. Por esta razão, é distribuído como código fonte, e deve ser compilado para o sistema operacional e plataforma específicos. Embora as funcionalidades sejam semelhantes em múltiplas plataformas, o processo de instalação é diferente. A seguir, alguns aspectos mais relevantes são ressaltados.

A plataforma deve atender alguns pré-requisitos para a instalação do *ARToolKit*. Em termos de hardware, é necessário suporte à entrada de imagens e um CPU que possa lidar com aquisição e processamento à taxa de trabalho especificada pelo programa usuário. A plataforma *Aramis* é provida de uma placa de captura *firewire*. Uma câmera *firewire Fire-i 400 Industrial Camera* foi utilizada como dispositivo de captura. Essas informações são relevantes para o processo de instalação, e por hora são suficientes para prosseguimento. As taxas de captura e de chamada das funções de estimação de pose são atendidas pela plataforma com folga. Na verdade, não se tira maiores proveitos de um aumento na taxa captura das imagens ou na taxa com que as poses são estimadas, pois o gargalo, como será visto, está na visualização, ou não visualização, de um alvo de rastreamento, e não no número de medidas que um dado alvo oferece. Mais detalhes sobre a câmera e sua montagem serão abordados posteriormente. A tabela 3.1 adiante, traz informações disponíveis no site do laboratório de tecnologia de interface humana da Universidade de Washington (HITL - *Human Interface Technology, University of Washington*), mostra alguns pré-requisitos de software necessários à instalação do *ARToolKit*.

Tabela 3.1: Pré-requisitos de software

Requisito	**Biblioteca Utilizada**	**Versão**
OpenGL e *GLUT*	*freeglut3-dev*	*2.6.0-0*
Biblioteca de Vídeo	*libraw1394-5*	*0.10.1-1*
Biblioteca de Vídeo	*libraw1394-dev*	*0.10.1-1*
Biblioteca de Vídeo	*libdc1394-13*	*1.1.0-3*
Biblioteca de Vídeo	*libdc1394-13-dev*	*1.1.0-3*

Apenas dois aspectos fogem aos procedimentos habituais de instalação. Embora não seja detalhado na documentacao padrão do *ARToolKit*, mesmo em suas versões mais atuais, a biblioteca faz uso de módulos de controle para câmeras digitais que utilizam o padrão *ieee1394* que não fazem mais parte das versões mais recentes dos respectivos pacotes nos repositórios padrão. Mais especificamente, o código do *ARToolKit* referencia funções que são parte do arquivo *libdc1394_ control.h*. O arquivo de biblioteca compartilhada *libdc1394_ control.so*, por sua vez, figura apenas nas versões 1.*x* de *libdc1394 (library for digital cameras ieee1394)*. A regressão da versão do software em questão envolve a regressão de uma cadeia bibliotecas relativas a captura pelo padrão *ieee1394*. A tabela 3.1 resume as versões das bibliotecas utilizadas.

Como se está instalando diferentes versões de bibliotecas compartilhadas, deve-se observar se os links simbólicos no diretório das bibliotecas apontam para os arquivos adequados à compilação. Nessa mesma linha, embora não se faça uso de recursos gráficos na aplicação, deve-se ajustar os links para bibliotecas de vídeo X na plataforma, para que seja possível compilar o *ARToolKit* com o recurso de compilação padrão fornecido com a biblioteca, sem que haja necessidade de se fazer alterações no *Makefile*.

Ainda sobre a instalação do *ARToolKit*, embora não se aplique diretamente à plataforma *Aramis* (que não possui até o momento uma versão muito recente do *kernel* do Linux), um comentário especial sobre os drivers *firewire* se faz necessário. A partir de 2010, uma nova pilha de *driver* de *kernel firewire* se tornou completamente funcional e pronta para substituir a pilha antiga *ieee1394*. Alguns problemas de compatibilidade podem ocorrer com aplicações que fazem uso de versões antigas de bibliotecas *firewire* como e o caso do *ARToolKit*. A princípio a mudança desses *driver* de *kernel* deveria ser transparente para bibliotecas de mais alto nível como é o caso da biblioteca *libraw1394*. De fato, se estão disponíveis apenas versões antigas da pilha de *drivers* no sistema, ou mesmo se têm-se ambas as versões disponíveis, a biblioteca *libraw1394 v2* pode selecionar a pilha de drivers adequada disponível. Porém, como já foi colocado, o *ARToolKit* utiliza versões 1 de bibliotecas de acesso *firewire*, como *libraw1394 v1*, por exemplo. Sendo assim, deve-se forçar que a pilha de *drivers* antiga seja habilitada em detrimento da mais nova. Essa configuração é necessária para utilizar o *ARToolKit* em máquinas em que as duas pilhas de *drivers* de *kernel* foram compiladas juntas. Para tanto deve-se editar um arquivo de configuração do sistema (*/etc/modprobe.conf*) adicionando as linhas:

```
#blacklist firewire-ohci
#blacklist firewire-sbp2
#blacklist firewire-net

blacklist ohci1394
blacklist sbp2
blacklist eth1394
blacklist dv1394
blacklist raw1394
blacklist video1394
```

Mais do que isso, a instalação de ambas as pilhas ao *kernel* somente é permitida para versões de *kernel* inferiores a 2.6.36 (inclusive) como uma forma de auxílio para o processo de migração. Ao tentar instalar um software como o *ARToolKit*, configurando sua compilação para funcionalidades *firewire*, numa distribuição mais atual do Linux, como Ubuntu 11.04, por exemplo, é necessário realizar uma regressão da versão do sistema, pois o *kernel* padrão da distribuição já é superior a 2.6.36 e, sendo assim, apenas será possível utilizar *libraw1394 v2* a menos que se recompile o *kernel* forçando a pilha de *drivers* de *kernel firewire* antiga.

3.3.2 Sistema de Coordenadas

Para estimação de posição e orientação de alvos de rastreamentos com respeito ao sistema de coordenadas da câmera, o *ARToolKit* define um sistema de coordenadas em cada marca e na câmera. Dado um determinado alvo, seu sistema de coordenadas tem origem no centro da figura, o eixo x cresce da esquerda para a direita e o eixo y cresce de baixo para cima. Seguindo a regra da mão direita, o eixo z parte da superfície da impressão e cresce na direção em que está impressa a figura. A Fig. 3.8 (adaptada da documentação online do *ARToolKit*) mostra um esquema de coordenadas sobre uma marca.

O sistema de coordenadas da câmera segue o padrão clássico de visão computacional. O centro cartesiano fica sobre o centro da imagem (obtido durante o procedimento de calibração, como será explicado adiante) e seus eixos x e y crescem na mesma direção em que crescem os índices horizontal e vertical da imagem, respectivamente. Novamente, pela regra da mão direita, o eixo z da câmera parte da lente e aponta para a região exterior à câmera. A Fig. 3.9 (adaptada da documentação online do *ARToolKit*) mostra a câmera e seu sistema de coordenadas ilustrado.

Mais adiante, quando a montagem física do sistema no laboratório for tratada, o sistema de coordenadas do laboratório (ou sistema de coordenadas absoluto) será definido, pois o sistema de localização com realidade aumentada como um todo levará em conta esses três sistemas de coordenadas: câmera, alvo (um sistema em cada alvo) e laboratório. Para formalização das equações de interesse do *ARToolKit*, deve-se ter em mente apenas esses dois sistemas já definidos (alvo, câmera), enfatizando o fato de que o *ARToolKit* não utiliza, em momento algum, qualquer informação externa. De fato, para o *ARToolKit*, a orientação ou posição do par câmera-alvo com respeito ao mundo é transparente, importando apenas a posição relativa entre essas duas entidades.

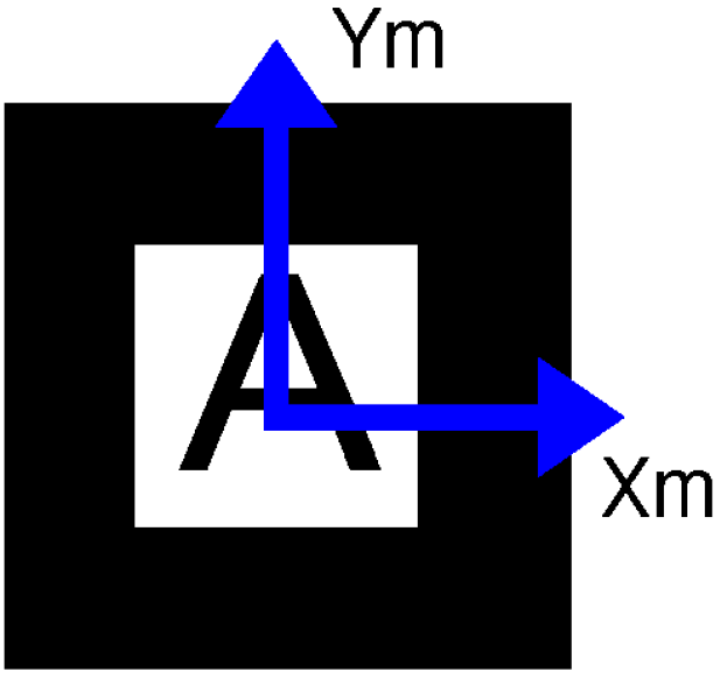


Figura 3.8: Sistema de coordenadas em cada alvo

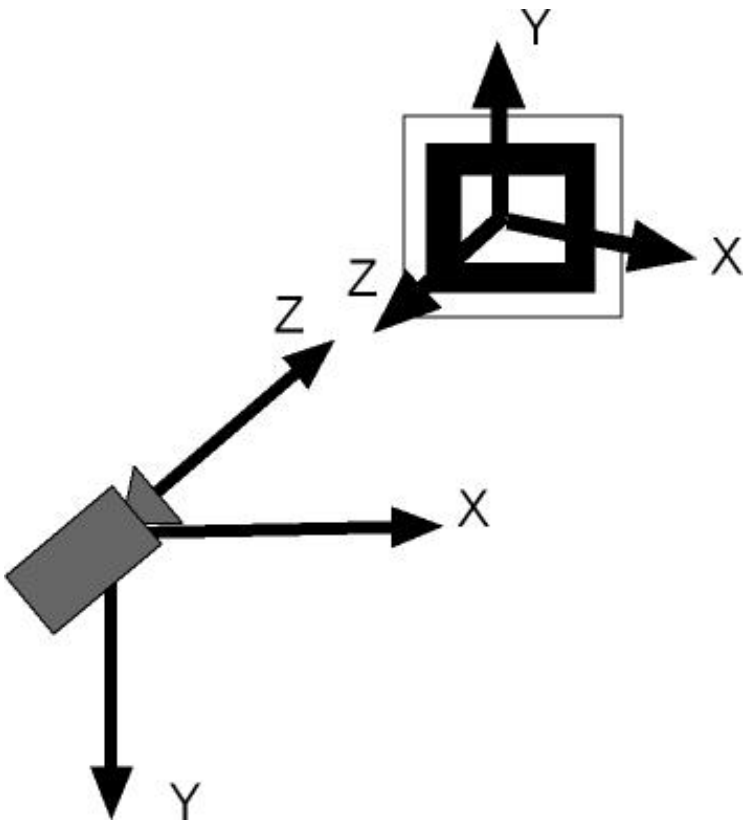


Figura 3.9: Sistema de coordenadas da câmera

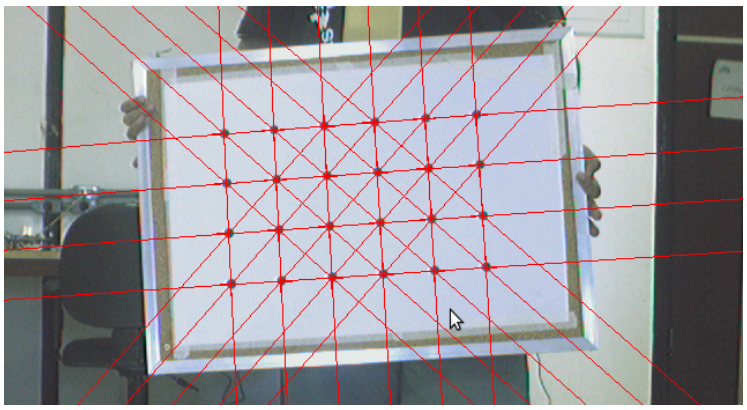


Figura 3.10: Padrão para calibração de distorções

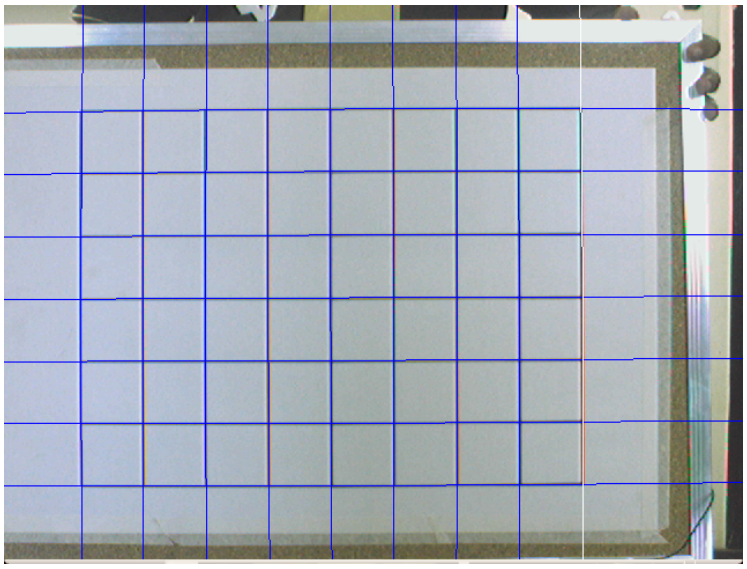


Figura 3.11: Padrão de calibração para aplicações 3-D

3.3.3 Calibração

Para o funcionamento do algoritmo do *ARToolKit* e necessário conhecer como pontos vistos pela câmera são mapeados em cada quadro. Em outras palavras, precisa-se conhecer os parâmetros intrínsecos da câmera através de um procedimento de calibração. Para alcançar esse objetivo utilizam-se padrões gráficos de dimensões conhecidas. Um primeiro procedimento leva em conta um padrão como o mostrado na Fig. 3.10. Esse procedimento visa recuperar aspectos como distorção e centro da imagem. A realização desta calibração é simples e suficiente para a maioria das aplicações de realidade aumentada. Os aspectos relativos ao funcionamento interno do *ARToolKit* aqui presentes, são descritos em [24].

Para aplicações em que se deseja obter medidas tridimensionais com o auxílio de *ARToolKit*, um procedimento mais complexo é requerido após o procedimento citado acima. O procedimento faz uso de uma figura quadriculada como visto na Fig. 3.11.

Nesse procedimento há interferência direta do usuário, que fica responsável por posicionar o

alvo em coordenadas específicas diante da câmera para coleta de dados. Como a câmera foi utilizada a cerca de 1.8 m dos alvos, buscou-se calibrá-la em torno dessa faixa de operação. Para isso aumentou-se o tamanho dos padrões de calibração para que tomassem uma porção apreciável da imagem quando na distância de trabalho.

Como mostra a Eq. 2.1, é possível entender uma transformação de coordenadas como uma matriz de rotação mais um vetor de translação. Esse conceito será utilizado para o desenvolvimento das equações a seguir, que traduzem as relações entre coordenadas.

$$\begin{bmatrix} hx_c \\ hy_c \\ h \\ 1 \end{bmatrix} = P \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{3.2}$$

Na Eq. 3.2 observa-se a matriz P de calibração, que traduz os pontos (X_c, Y_c, Z_c) descritos no sistema de coordenadas da câmera para pontos (x_c, y_c) de coordenadas no quadro da câmera. Pode-se reescrever Eq. 3.2 como função das coordenadas (X_t, Y_t, Z_t), descritas no sistema de coordenadas de um alvo de rastreamento (*target*, em inglês) se for acrescentada a transformação de T_c^t.

$$\begin{bmatrix} hx_c \\ hy_c \\ h \\ 1 \end{bmatrix} = PT_c^t \begin{bmatrix} X_t \\ Y_t \\ Z_t \\ 1 \end{bmatrix} \tag{3.3}$$

As matrizes P e T juntas compõem a matriz C que é a matriz que será realmente estimada pelo algoritmo de calibração. Pois dispomos de vários pares de coordenadas (x_c, y_c) e (X_t, Y_t, Z_t), dado que o procedimento de calibração posicionou o alvo nas posições previamente informadas ao software de calibração.

$$\begin{bmatrix} hx_c \\ hy_c \\ h \\ 1 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_t \\ Y_t \\ Z_t \\ 1 \end{bmatrix} \tag{3.4}$$

As matrizes T e P são da forma:

$$P = \begin{bmatrix} s_x f & 0 & x_0 & 0 \\ 0 & s_y f & y_0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; T_c^t = \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{13} & T_y \\ R_{31} & R_{32} & R_{13} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Onde P é a matriz de transformação de perspectiva, na qual se tem interesse. Em P, s_x e

s_y são fatores de escala $[pixel/mm]$ nas direções x e y respectivamente, e f é a distância focal. O ponto (x_0, y_0) representa a posição em que o eixo Z_c passa pelo quadro. Uma vez que se obtém apenas a matriz C, ela é decomposta em duas matrizes da forma de P e T. Para que a decomposição determine a matriz P, é necessário que o número de elementos independentes em P mais o número de elementos independentes em T seja igual ao número de elementos independentes em C. C possui 11 elementos independentes, enquanto P e T juntas possuem apenas 10. Por isso, é acrescentado um elemento a mais (k) na matriz P. Este elemento, embora tenha sido introduzido explicitamente para possibilitar a decomposição, possui um significado físico. Ele está relacionado ao ângulo entre os eixos x e y. Idealmente seu valor seria zero, o que significaria eixos x e y perfeitamente ortogonais. Espera-se portanto que o valor deste termo, após estimação seja próximo de zero.

São substituídos então, valores de (x_c, y_c) e (X_t, Y_t, Z_t) na Eq. 3.4 para se estimar os termos e depois decompor as matrizes. A equação final é da forma de Eq. 3.6.

$$\begin{bmatrix} hx_c \\ hy_c \\ h \\ 1 \end{bmatrix} = \begin{bmatrix} s_x f & k & x_0 & 0 \\ 0 & s_y f & y_0 & 0 \\ 0 & 0 & 01 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} & R_{13} & T_x \\ R_{21} & R_{22} & R_{13} & T_y \\ R_{31} & R_{32} & R_{13} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_t \\ Y_t \\ Z_t \\ 1 \end{bmatrix} \tag{3.6}$$

No fluxo de trabalho normal do *ARToolKit*, quando um padrão é reconhecido, uma matriz de transformação é retornada com *arGetTransMat*(), que define posição e orientação relativa (6 graus de liberdade) do marcador para a câmera, no sistema de coordenadas da câmera calibrada. Dados de calibração da câmara são usados por essa função de forma a modificar os resultados para compensar para as propriedades intrínsecas da câmera. Devido a distorções na câmera e erros no processo de calibração, o sistema de coordenadas tridimensional retornado não é um sistema de coordenadas ortogonais, tipicamente utilizado em gráficos 3D [25].

Se esta matriz retornada é usada como sistema de coordenadas da câmera para desenhar um objeto com *libARgsub*, a perspectiva de visão usados pelo *OpenGL* será a mesma que a do modelo da câmera, e assim a imagem parecerá renderizada no local correto. Um interessante efeito dessas transformações é que não importa quão pobre a calibração da câmara é, o objeto 3D sobreposto no marcador (motivacao típica da realidade aumentada) será sempre correta, já que a calibragem incorreta é revertida ao desenhar utilizando a câmera como perspectiva de vista em *libARgsub* [25]. Este efeito na verdade, é muito positivo para aplicações típicas de realidade aumentada por visão, pois se traduz numa maior robustez do processo de sobreposição de imagens.

Assim, aplicações menos exigentes do *ARToolKit* alcançam resultados com uma calibração de distorção simples, enquanto aplicações que necessitam de maior acurácia de medidas tridimensionais devem fazer uso do processo de calibração que utiliza uma grade de linhas posicionadas em posições controladas. Esse procedimento de calibração, embora busque fazer uma estimativa robusta dos parâmetros de interesse, é sujeito a erros dado a interferência direta do usuário no posicionamento da câmera e do padrão de calibração. A Fig. 3.12 (adaptada da documentação online do *ARToolKit*) traz um esquema do processo de calibração descrito.

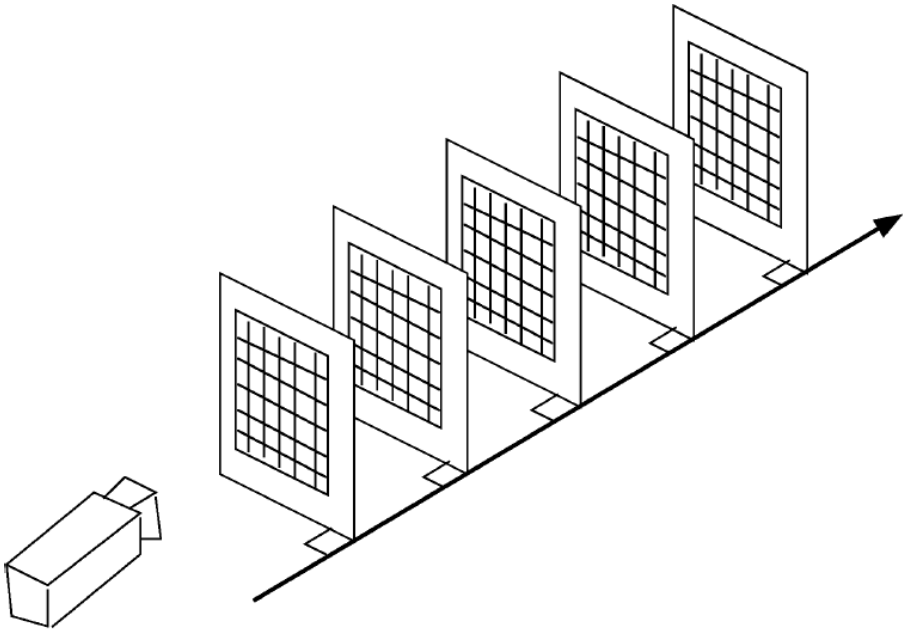


Figura 3.12: Representação de procedimento de calibração. A cargo do usuário, o padrão deve ser posicionado de forma acurada em posições e orientações espaciais bem definidas diante da câmera. (**fonte**: adaptada da documentação online do *ARToolKit*)

De forma a reduzir os erros de medida dados pelo *ARToolKit*, foi proposto um filtro estendido pelos parâmetros de calibração, descrito pela Eq. 3.7, que traduz a evolução de pose e parâmetros, e pela Eq. 3.8 que representa o passo de medição.

A dinâmica do filtro foi implementada da seguinte forma: a cada instante de amostragem das velocidades das rodas, o sistema evoluía pela equação de processo Eq. 3.7. O passo de correção do filtro se dava conforme sugerido na Eq. 3.8. Sempre que se dispunha de um marcador, era utilizado o *ARToolKit* para obter a posição e orientação da câmera. A cada correção, era estimada a posição da câmera e realimentava-se os parâmetros utilizados para gerar as medidas (parâmetros de calibração) com f_{ARTK} para a próxima iteração. Dessa forma, o algorítimo do filtro deveria ser desenvolvido durante a coleta de dados com o robô, pois precisava-se alterar os parâmetros de calibração do *ARToolKit* a cada correção, e obter dados com parâmetros atualizados. Assim o filtro descrito por Eq. 3.7 e Eq. 3.8 foi implementado em linguagem *C* dentro da plataforma Aramis. Uma estratégia de comunicação entre os módulos de filtro e aquisição de imagens teve de ser desenvolvida, pois antes de cada aquisição e processamento com o *ARToolKit* era necessário checar por uma possível mudança nos parâmetros de calibração e carregar os novos parâmetros em caso positivo.

$$\begin{bmatrix} x_{rob,k+1} \\ y_{rob,k+1} \\ \theta_{rob,k+1} \\ \delta x_{k+1} \\ \delta y_{k+1} \\ s_x f_{k+1} \\ k_{k+1} \\ x_{0,k+1} \\ s_y f_{k+1} \\ y_{0,k+1} \\ x_{c,k+1} \\ y_{c,k+1} \\ dist_{factor,k+1} \\ size_{adjust,k+1} \end{bmatrix} = \begin{bmatrix} x_{rob,k} \\ y_{rob,k} \\ \theta_{rob,k} \\ \delta x_{k} \\ \delta y_{k} \\ s_x f_{k} \\ k_{k} \\ x_{0,k} \\ s_y f_{k} \\ y_{0,k} \\ x_{c,k} \\ y_{c,k} \\ dist_{factor,k} \\ size_{adjust,k} \end{bmatrix} + \begin{bmatrix} cos\,(\theta) & -sen\,(\theta) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ sen\,(\theta) & cos\,(\theta) & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{v_r+v_l}{2} \\ 0 \\ \frac{-v_r+v_l}{2l} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.7}$$

$$\begin{bmatrix} x_{rob,k+1} \\ y_{rob,k+1} \\ \theta_{rob,k+1} \end{bmatrix} = f_{ARTK}(param) \tag{3.8}$$

Por mais que se tenha tentado modelar a correlação entre os parâmetros e a estimativa gerada, o equacionamento não era suficiente para uma atualização correta dos parâmetros e aprofundar-se nesse sentido fugia ao escopo do trabalho. O desenvolvimento do filtro citado mostrou-se uma má decisão de projeto no caso específico de interesse deste trabalho e, não gerando resultados satisfatórios, sua depuração e aprimoramento foram abandonados. Mesmo assim, há trabalhos que sugerem alguma melhora pela utilizacao de alto-calibração para *ARToolKit* [26] e uma investigação mais profunda nesse sentido pode ser sugerida como um trabalho futuro.

Figura 3.13: Fixação da câmera no *Aramis*

3.3.4 Montagem Física do Sistema

A montagem física do sistema consiste primeiramente na definição de um sistema de coordenadas para posicionamento dentro do laboratório. Em seguida se dá o posicionamento das marcas em posições conhecidas dentro do laboratório e montagem da câmera junto ao robô.

Para determinação de um sistema de coordenadas no laboratório foi definida uma origem, uma direção para o eixo x e outra para o eixo y. A escolha desses eixos poderia ser feita de forma arbitrária. Na verdade, a definição utilizada já vinha sendo utilizada em trabalhos anteriores com localização com RFID no LARA. Como foram realizados testes ainda neste trabalho, foram mantidas as considerações. A Fig. 3.1 ilustra a definição dos eixos e origem. O eixo z, seguindo a regra da mão direita, aponta para baixo.

No *Aramis*, a câmera foi posicionada buscando estar ao máximo alinhada com seu eixo principal x e apontando para cima na direção do eixo z (vertical). Possíveis variações na posição da câmera (como uma angulação com relação a vertical, um deslocamento com respeito ao robô, e outros) se traduzem em medidas erradas com a utilização do *ARToolKit*. Embora saiba-se que esses possíveis erros de montagem geram na verdade um bias nas medidas, essas variações são, por simplicidade, consideradas como aleatoriedade do sistema e vão ser contabilizadas no ruido de medição, que será abordado adiante.

As marcas foram fixadas no teto. Essa escolha traz dois fatores desfavoráveis ao sistema. Em primeiro lugar, a distância do topo do robô (local onde foi fixada a câmera) até o teto é muito grande e potencializa os erros cometidos pela estimação de orientação das marcas, como será visto adiante. Por último, a câmera fica diretamente apontada para a fonte de luz, o que prejudica a identificação de formas e localização de cantos pelo algoritmo computacional. Por simplicidade, foi convencionado a fixação das marcas com os eixos x e y alinhados com os eixos x e y do laboratório. Assim, dada a pose da câmera com respeito a uma marca, obtém-se a posição no laboratório pela simples translação de coordenadas correspondente a posição em que a marca foi fixada.

Aspectos relativos ainda a montagem do sistema, no que diz respeito a escolha de uma es-

Figura 3.14: Trecho com alvos no teto

tratégia para identificação e localização de alvos, levam em conta características mais profundas do funcionamento do software. Adiante, será retomado esse tema, uma vez que já se tenha um entendimento melhor do algoritmo para identificação de marcas.

3.3.5 Fluxo de Trabalho

O segredo para a sobreposição de imagens (funcionalidade principal do *ARToolKit*) está nos quadrados pretos presentes nos alvos utilizados como marcadores de rastreamento. De forma simplificada, o rastreamento do *ARToolKit* funciona da seguinte forma:

1. A câmera realiza captura de vídeo do mundo real e envia para o computador.
2. Software no computador busca, em cada quadro do vídeo, por quaisquer formas quadradas.
3. Se um quadrado é encontrado, o software usa um desenvolvimento matemático para calcular a posição da câmera em relação ao quadrado preto.
4. Uma vez que a posição da câmera é conhecida um modelo gráfico computacional é desenhado a partir dessa mesma posição.
5. Este modelo é desenhado em cima do vídeo do mundo real, e assim aparece preso no marcador quadrado.

O resultado final é mostrado no dispositivo de saída, assim quando o usuário olha através do visor vê gráficos sobrepostos ao mundo real. Os passos 1 a 3 descritos acima são os que são úteis no sistema de localização.

3.3.5.1 Captura

No programa principal do robô Aramis, um módulo é definido para realizar todo o processamento de imagens necessário ao sistema de localização. Este módulo corresponde aos arquivos *camera.cpp* e *camera.h*, que por sua vez faz uso de um módulo auxiliar para definir e instanciar marcas de rastreamento e suas características, correspondendo aos arquivos *patterns.cpp* e *patterns.h*. O módulo de captura define uma tarefa em tempo real no Linux com Xenomai que está encarregada da captura e processamento de toda informação de localização a partir das imagens, quadro a quadro. Esta tarefa é definida periódica e ocorre a períodos de aproximadamente 0.5 s. O *ARToolKit* recupera parâmetros de configuração de vídeo de uma cadeia de caracteres com definições que deve ser montada segundo direcionamento fornecido pelo próprio software.

Os únicos atributos definidos na aplicação foram o modo e a taxa de captura. Quanto ao modo, deseja-se a maior resolução possível na imagem, pois será utilizada posição de cantos identificados na matriz de *pixels* da imagem para inferir posição e orientação da câmera. Com a câmera disponível ao projeto, a maior resolução alcançada é 640×480 *pixels*. Os modos disponíveis restringem-se então a *640x480_RGB* e *640x480_411YUV*, tendo sido escolhido *640x480_RGB*. Quanto à taxa de captura, obviamente seria interessante ter quantas medidas fossem possíveis. Porém a tarefa de captura está limitada por um período de meio segundo pois leva-se em conta o processamento das imagens recebidas, o que limita a utilização em cerca de 2 frames por segundo. A taxa foi definida como 7.5 quadros por segundo, pois como será visto adiante, O gargalo da localização não está na taxa em que se computa uma imagem vista, mas sim no tempo em que não se adquire nenhuma imagem de um alvo de rastreamento (inerente a montagem física do experimento.)

3.3.5.2 Identificação de Alvos

De posse de uma imagem, o *ARToolKit* se encarrega de identificar quadriláteros na imagem. O software aplica um limiar à imagem, diferenciando entre *pixels* pretos e brancos. Então, através de algoritmos de visão computacional, busca por regiões com contorno que possam ser definidas por quatro linhas com baixo erro de ajuste. Este trecho do algoritmo merece destaque devido a uma nuance da montagem física do sistema. Em uma matriz de *pixels* de uma determinada imagem, valores elevados significam que um *pixel* esteve mais exposto a luz, enquanto valores baixos indicam baixa exposição. Se for aumentando progressivamente a intensidade de luz no ambiente, *pixels* que antes apresentavam valor baixo começam a aumentar seu valor, enquanto *pixels* que antes já tinham valor alto mantém-se estagnados no valor máximo. Este fenômeno de saturação aproxima os valores numéricos de pixels claros e escuros para regiões de iluminação exagerada ou para regiões da imagens voltadas a fontes de luz, que saturam os *pixels* ao seu redor. Esse efeito pode afetar a identificação de cantos nas marcas utilizadas pelo *ARToolKit*, pois diminui

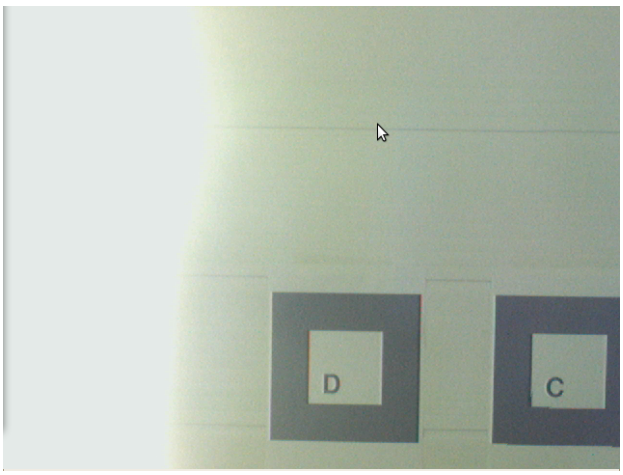


Figura 3.15: Padrão não identificado por luminosidade inadequada

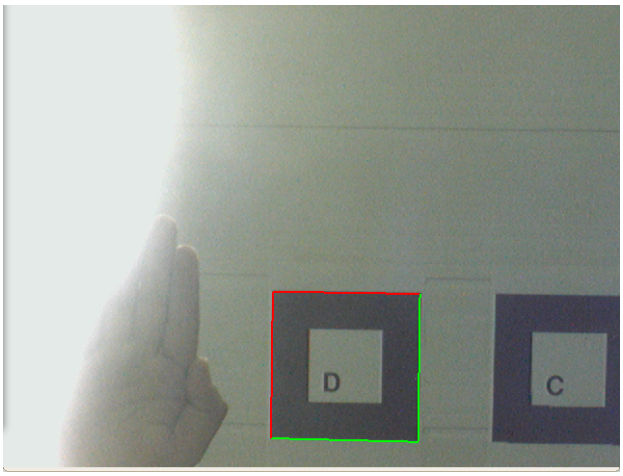


Figura 3.16: Padrão identificado

a diferença de valores entre *pixels* claros e escuros. Um exemplo claro dessa dificuldade é ilustrado nas Fig. 3.15 e Fig. 3.16.

As Fig. 3.15 e 3.16 foram obtidas utilizando o utilitário *mk_patt* fornecido com o *ARToolKit*. Este programa é um utilitário para criar novos alvos a serem fornecidos a outros programas. O *mk_patt* utiliza o mesmo algoritmo de busca de regiões definidas por quadriláteros do *ARToolKit* que é fornecido em sua interface (API). Este utilitário busca por *patterns* nas imagens e marca seu contorno como identificado. O usuário deve então confirmar que o contorno foi identificado e inserir um nome para o arquivo que guardará informações sobre aquele alvo de rastreamento. A saber, o arquivo guarda um *bitmap* do padrão identificado, que será posteriormente comparado por correspondência de *templates* aos alvos identificados por aplicações terceiras. Observa-se que, devido à intensidade da luz, o contorno não é encontrado com ajuste de linhas satisfatório pelo *software*. Ao bloquear parte da incidência com uma das mãos, o contorno é satisfatoriamente encontrado.

Uma vez encontrado um contorno em uma imagem, a região interna ao contorno deve ser

Figura 3.17: Processo de normalização e redução de resolução(**fonte**: adaptada de documentação online do *ARToolKit*)

normalizada para ser comparada a padrão de alvos fornecidos previamente ao programa (esses padrões são criados com o aplicativa já citado *mk_ patt*). Para essa normalização e utilizada uma transformação de perspectiva mostrada na Eq. 3.9. Os termos N_{ij} na matriz de transformação são obtidos pela substituição direta de coordenadas dos vértices do quadrilátero na equação. Do lado esquerdo, são substituídas as coordenadas do vértice no quadro da imagem (x_c, y_c), enquanto do lado direito, são colocadas as coordenadas dos vértices no sistema de coordenadas do alvo (X_m, Y_m).

$$\begin{bmatrix} hx_c \\ hy_c \\ h \end{bmatrix} = \begin{bmatrix} N_{11} & N_{12} & N_{13} \\ N_{21} & N_{22} & N_{13} \\ N_{31} & N_{32} & N_{13} \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \end{bmatrix} \tag{3.9}$$

A equação de normalização pode ser aplicada a todos os pontos internos à região da subimagem definida pelo contorno.

De certa forma, o processo de normalização é custoso. Sendo assim, recomenda-se evitar que numa mesma imagem haja um número elevado de alvos. Para o sistema de localização proposto, não mais do que duas marcas no teto são vistas por vez. Em primeiro lugar, a câmera possui uma grande distância focal, gerando um cone de visão mais fechado. Isto, além de limitar o número de marcas completamente observadas numa imagem, aumenta o número de *pixels* que descreve uma marca, dada a resolução fixa em 640×480. Outro fator que reduz a chance de mais imagens aparecerem num mesmo quadro é a escolha de alvos com tamanho elevado. Essa escolha parte de uma análise de desempenho quanto a acurácia do *ARToolKit* que será vista adiante.

Após o processo de normalização, ainda é feita uma redução de resolução. Esta redução tem por objetivo facilitar computacionalmente o *template matching* entre a subimagem normalizada e os *bitmaps* fornecidos previamente ao programa. Neste sentido, deve haver um compromisso entre qualidade e velocidade da identificação do padrão. Na aplicação para o sistema de localização de robótica móvel, era buscado reduzir os custos computacionais da tarefa de sensoriamento com a câmera. Uma ilustração é apresentada na Fig. 3.17 (adaptada de documentação online do *ARToolKit*).

O trecho de código do arquivo *config.h* do *ARToolKit*, apresentado a seguir, traz definições que podem ser alteradas conforme a aplicação de interesse.

```
1 #define   AR_SQUARE_MAX        30
2 #define   AR_CHAIN_MAX      10000
3 #define   AR_PATT_NUM_MAX      50
4 #define   AR_PATT_SIZE_X       16
5 #define   AR_PATT_SIZE_Y       16
6 #define   AR_PATT_SAMPLE_NUM   64
```

No trecho, as definições *AR_PATT_SIZE_X* e *AR_PATT_SIZE_Y* refletem a resolução da subimagem após a redução. A princípio, o sistema de localização para robótica móvel em questão foi pensado com 70 marcadores diferentes, e para tanto, foi alterado o valor da definição *AR_PATT_NUM_MAX*. Foram utilizados símbolos como letras do alfabeto e outros caracteres, ocupando sempre o quadrante inferior esquerdo (x e y negativos no sistema de coordenadas do alvo) de forma a evitar simetrias. É claro que imagens simétricas tornariam impossível para sistema determinar informação de posição e orientação adequada.

No entanto, essa grande variabilidade de imagens torna impossível a identificação do padrão correto sem que se aumente a resolução definida por *AR_PATT_SIZE_X* e *AR_PATT_SIZE_Y*. Entretanto, essa escolha de figuras so apresentou resultados próximos ao satisfatório para um numero muito elevado na resolução final da subimagem, o que se imaginou ter efeito negativo na performance, dado o elevado número de *patterns* a se comparar. Embora não se tenha feito um estudo quantitativo do aumento da complexidade e custo computacional para avaliar a viabilidade desta solução, a linha geral para decisões de projeto aqui sempre visam reduzir custos computacionais, pois não se sabe que sistemas serão implementados na plataforma Aramis em tempo real no futuro, e que estariam dividindo recursos com a tarefa de localização. Sendo assim, uma nova estratégia foi utilizada. Manteve-se a resolução original da biblioteca (16×16). Em contrapartida, reduziu-se o número de figuras de 70 para 11, de tal forma que as 11 figuras alcançassem a simetria utilizando uma porção maior da subimagem aumentando a diferenciação entre as figuras e facilitando sua correspondência mesmo com a resolução baixa. Dessa forma, para que se pudesse cobrir toda a área de interesse do laboratório com os alvos de rastreamento, as figuras foram repetidas em diferentes locais (foram geradas entre 3 e 4 figuras de cada tipo) de modo que uma mesma figura ficasse o mais espalhada possível no ambiente. A seguir, foi implementado um algoritmo para considerar aquela figura que estivesse coordenadas de menor distância para as coordenadas estimadas pelo sistema de localização em si do robô. Se, por exemplo, existisse uma figura definida por um arquivo de *pattern* em cada quadrante do laboratório, e o robô reconhecesse uma delas, tomaria as coordenadas daquela que estivesse no mesmo quadrante em que se estimasse estar o robô. Uma consequência direta dessa estratégia, e que o robô deve começar numa posição inicial conhecida, pois ao identificar um padrão, utilizará sua informação prévia de posição para interpretar o sensor.

Foi construída uma estrutura de dados para os alvos, definida no módulo descrito pelos arquivos *pattern.h* e *pattern.cpp*. Cada instância da estrutura guarda um nome e índice associado a uma determinada figura, bem como o número de vezes em que essa figura aparece no laboratório (limitado a 10) e quais as posições de cada ocorrência da figura. O trecho de código abaixo mostra a definição da estrutura e um exemplo de instância para uma determinada figura.

```
typedef struct
{
    int target_id;
    int nCopias; //nro de copias do mesmo pattern
    double target_center_lab_x[MAX_COPIAS];
    double target_center_lab_y[MAX_COPIAS];
    double target_width;
    char pattname[256];
} PATTERN;

strcpy(patts[10].pattname,"Data/patts/patt.inang");
    patts[10].nCopias=3;
    patts[10].target_center_lab_x[0] = 2680.0;//ok
    patts[10].target_center_lab_y[0] = 5680.0;//ok
    patts[10].target_center_lab_x[1] = 3500.0;//ok
    patts[10].target_center_lab_y[1] = 8760.0;//ok
    patts[10].target_center_lab_x[2] = 3490.0;//ok
    patts[10].target_center_lab_y[2] = 960.0; //ok
    patts[10].target_width = SW;
```

Com essa arquitetura de solução, foram realizados vários testes quanto a possibilidade de o sistema reconhecer um *patterns* de maneira inadequada, confundindo entre figuras, ou dado o reconhecimento de uma figura, quanto a possibilidade de o sistema recuperar as coordenadas inadequadas da instância da figura sendo observada. Em geral foram necessários apenas algumas alterações nas próprias figuras desenhadas dentro das marcas, pois embora o sistema diferenciasse corretamente entre figuras e fornecesse coordenadas corretas, muitas vezes a orientação retornada era errada. Ou porque a figura realmente possuía alguma simetria que escapava aos olhos no momento de sua concepção, ou porque as nuances de assimetria não estavam sendo bem identificadas pelo *ARToolKit*. Feitos os ajustes o sistema funcionou corretamente, sem cometer qualquer erro apos um teste de cerca de 7 minutos navegando por toda a extensão do laboratório repetidas vezes.

No laboratório, percebe-se que há várias regiões desprovidas de marcação. Essas regiões ocorrem por duas razões distintas. Num primeiro caso, a região estaria sobre obstáculos de forma que o robô nunca passaria por sob esta região, sendo inútil o posicionamento de marcadores. Num segundo caso, há dificuldade de se fixar marcadores por obstáculos no próprio teto (tubulações, vigas de concreto, e outros). No último caso, mostra-se uma situação desfavorável ao sistema, pois o robô poderia trafegar por algum tempo sem ter a possibilidade de receber correções.

3.3.5.3 Recuperação da matriz de transformação

Para recuperação da matriz de transformação, apenas serão combinadas informações já obtidas até o momento. Pontos pertencentes a linhas de contorno de um marcador estão relacionados a suas coordenadas no sistema de coordenadas da câmera por meio da Eq. 3.2. As equações de linhas paralelas projetadas na imagem, por sua vez, já foram determinadas no processo de ajuste de linhas. Substituindo as coordenadas de pontos pertencentes às linhas do contorno do marcador nessas equações, por suas coordenadas no sistema de coordenadas da câmera, pode-se

Tabela 3.2: Faixa utilizável de distância por tamanho de marcador considerado

Tamanho do padrão $[mm]$	**Faixa Utilizável** $[mm]$
69.85	406
99.90	635
107.95	834
187.20	1270

aplicar relações geométricas conhecidas dos alvos (linhas perpendiculares ou paralelas, e outros) e repetir o procedimento para cada conjunto de dados que se tem posse. Usa-se um critério de erro quadrático para estimar os parâmetros da matriz de transformação.

A derivação das equações aqui, não traz maiores benefícios. Apenas a noção de dependência dos resultados com a identificação de linhas, cantos e calibração são suficientes para o entendimento do trabalho.

3.3.6 Limitações

Como visto, o *ARToolKit* baseia-se em posições de *pixels*, correspondentes aos vértices das marcas, que foram mapeados na imagem. Usualmente, erros na localização de *pixels* ocorrem independentemente do tamanho da marca e estão mais relacionados à qualidade da câmera e do ambiente. Contudo, o erro de localização de um canto, por um determinado número de *pixels*, terá seu efeito de erro na orientação estimada do alvo cada vez menor quanto maior o tamanho da marca. Nesse sentido, deve-se utilizar marcas o quão grande possível. Porém, é necessário fazer um compromisso, pois com marcas exageradamente grandes, há uma probabilidade maior de a câmera passar por sobre a marca e nunca ter, num mesmo quadro, todos os cantos dentro do campo de visão. A Tabela 3.2 [24] mostra uma comparação dos erros tomados por marcadores quadrados de diferentes tamanhos. Estes resultados foram obtidos utilizando padrões de diferentes tamanhos (comprimento de um lado), colocando-os perpendicular à câmera e movendo a câmera para trás até que o algoritmo do *ARToolKit* parasse de identificar e calcular a posição relativa de um alvo.

A faixa de alcance também é afetada pela complexidade de padrões. Quanto mais simples o padrão melhor para o alcance. Padrões com grandes regiões em preto e grandes regiões em branco (ou seja, padrões de frequência baixa) são os mais eficazes. Com padrões de maior complexidade, os resultados de alcance na tabela 3.2 podem ser reduzidos para menos da metade. Os padrões escolhidos para o projeto, buscaram o mínimo de complexidade que permitisse uma diferenciação clara entre duas formas diferentes e a importante assimetria da figura. Uma listagem das imagens escolhidas e mostrada no anexo II.

O rastreamento também é afetado pela orientação relativa do marcador para a câmera. O rastreamento pode se tornar inviável para situações em que o marcador está demasiadamente inclinado com relação à linha de centro da câmera. Da montagem física do sistema em questão, espera-se que os alvos sejam vistos segundo pequenos desvios angulares, pois em geral a câmera está praticamente vertical para cima e as figuras sempre apontam seu eixo z na vertical para baixo.

Como já foi visto, os resultados também são afetados por condições de iluminação. Luzes do teto pode criar reflexões em pontos reflexivos em um marcador de papel e assim torná-lo inviável para o algoritmo. A documentação do *ARToolKit* sugere, para reduzir efeitos negativos de reflexão dos marcadores, sua impressão em material não-reflexivo. No projeto foi utilizada a impressora a laser sobre um papel A4 comum.

3.4 Sistema de Localização com Câmera

Tendo em vista as informações apresentadas sobre a ferramenta de realidade aumentada e as decisões de projeto feitas em função de suas características, iniciou-se a implementação de um sistema de localização. O sistema faz uso de informações de velocidade das rodas fornecidas pelos *encoders* da plataforma *Aramis* para compor um passo de predição por odometria utilizando um modelo cinemático de robô a tração diferencial. Com o uso das ferramentas de realidade aumentada, é recomposta a posição e orientação do robô a cada instante em que um quadro e recuperado da câmera e se tem um marcador válido identificado. As equações Eq. 3.10 e Eq. 3.11 são as equações de processo e medição do modelo do Filtro de Kalman *Unscented* que foi utilizado para combinar as informações de odometria e imageamento.

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} cos\left(\theta_{k-1}\right) & -sen\left(\theta_{k-1}\right) & 0 \\ sen\left(\theta_{k-1}\right) & cos\left(\theta_{k-1}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{v_r+v_l}{2} \\ 0 \\ \frac{-v_r+v_l}{2l} \end{bmatrix} \cdot dt \tag{3.10}$$

$$medida_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \tag{3.11}$$

O *ARToolKit* trabalha com milímetros e por isso a unidade de medida adotada para o sistema de localização foi milímetros para distâncias e radianos para ângulos. Na verdade, o que o *ARToolKit* realmente fornece, é a transformação (rotação e translação) entre a câmera e o alvo considerado. Nesse sentido, um modelo convencional da equação de medição no filtro consistiria numa equação que, dada uma posição estimada, retornasse a transformação associada ao alvo considerado. Como sugere a Eq. 3.11, é utilizado um modelo com medição inversa. Recupera-se do *ARToolKit* a transformação de coordenadas entre câmera e alvo, aplica-se ao ponto de centro do robô (descrito no sistema de coordenadas da câmera) e, em seguida, aplica-se a transformação entre sistema de coordenadas do alvo e sistema de coordenadas absoluto (apenas um deslocamento, por efeito da montagem do sistema). Com isso utilizam-se os dados fornecidos pelo *ARToolKit* para recompor a posição e orientação do robô móvel, e só então essa medida e utilizada no passo de correção do filtro, gerando um fator de inovação por comparação direta com o vetor de estados (posição e orientação).

Retomando a fundamentação teórica do trabalho, sabe-se que a cada equação está associada um ruido suposto gaussiano de média nula e matriz de covariâncias dadas $Q\left(k\right)$, para equação de

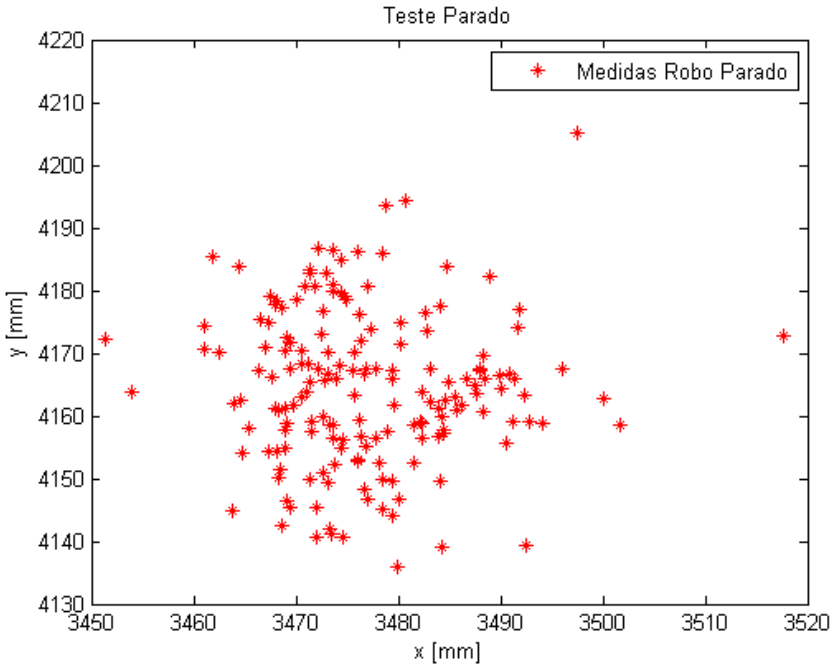


Figura 3.18: Posicionamento retornado pelo *ARToolKit* num experimento com robô parado

processo, e $R(k)$ para equação de medição. A Fig. 3.18 foi obtida conduzindo um experimento com o robô mantido imóvel sob uma marca no teto.

Os testes apontaram um desvio padrão maior em uma das direções. Contudo, ao se modificar a orientação do robô em 90 graus as direções com maior e menor desvio se invertiam. Além de os erros dependerem da posição relativa do robô com respeito a marca e a características físicas no ambiente como iluminação, foi observado que na realização de um movimento puramente circular sob uma marca ($v_r = -v_l$) o resultado era ligeiramente diferente do que o esperado em termos de localização, pois em vez de apontar uma trajetória circular conforme a que foi medida fisicamente durante a realização do experimento (raio da trajetória em torno de 18 cm), o *ARToolKit* fornecia uma trajetória circular com raio ligeiramente maior (em torno de 35 cm). No artigo [2] o autor chega a resultados semelhantes aos obtidos no projeto, repetindo um experimento em um ambiente bastante controlado. As Fig. 3.20 e Fig. 3.19 (adaptadas de [2]) mostram resultados publicados pelo autor que condizem com os resultados obtidos no sistema de localização aqui descrito. No artigo, o autor não chega a nenhuma conclusão concreta com respeito às causas dos fenômenos.

Embora não se tenha um estudo mais profundo das causas relacionadas, esse e outros efeitos podem estar relacionados a características da calibração, que modelam distorções com fatores de escala em cada direção. A má estimação de distorções e, em especial, uma má localização do centro óptico no quadro da imagem poderiam causar as anomalias citadas. Como já foi colocado, a calibração do *ARToolKit* não tem efeito crítico para aplicações de sobreposição de imagens. Para corrigir erros do sistema de calibração do *ARToolKit* [25] sugere uma alteração forçada dos parâmetros de calibração levando o centro óptico da imagem aos *pixels* centrais no quadro da câmera. Essa alteração não trouxe grandes melhoras, e seu embasamento está unicamente no fato

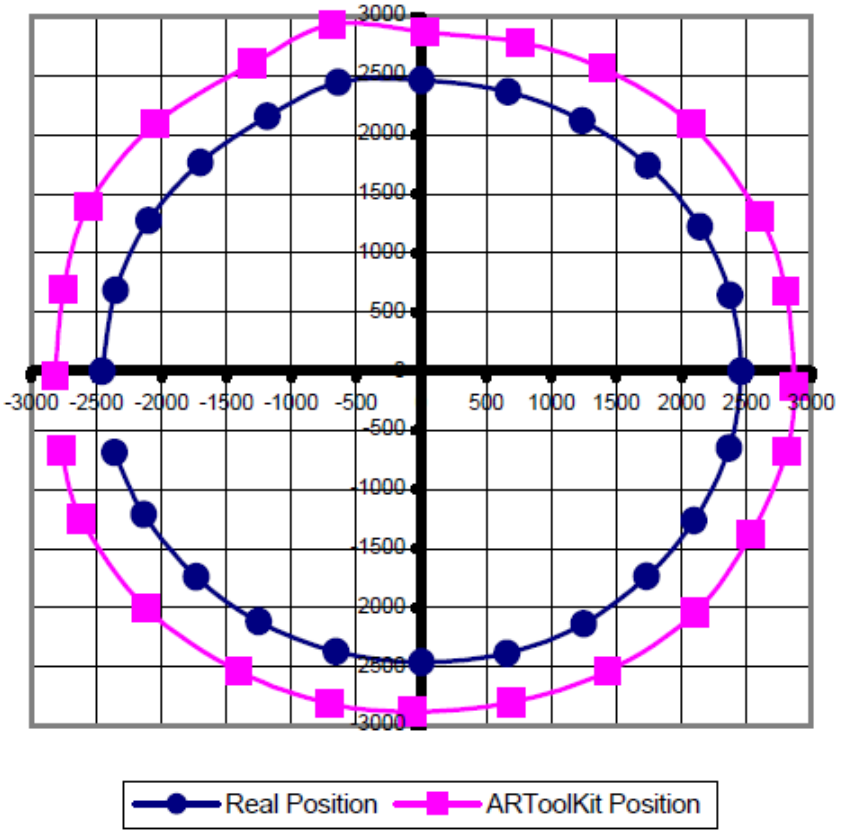


Figura 3.19: Em azul, medidas de raio de uma trajetória circular da câmera obtidas pelo *ARToolKit*. Em vermelho, medidas aferidas fisicamente com instrumentos de medição. (**fonte:** adaptada de [2])

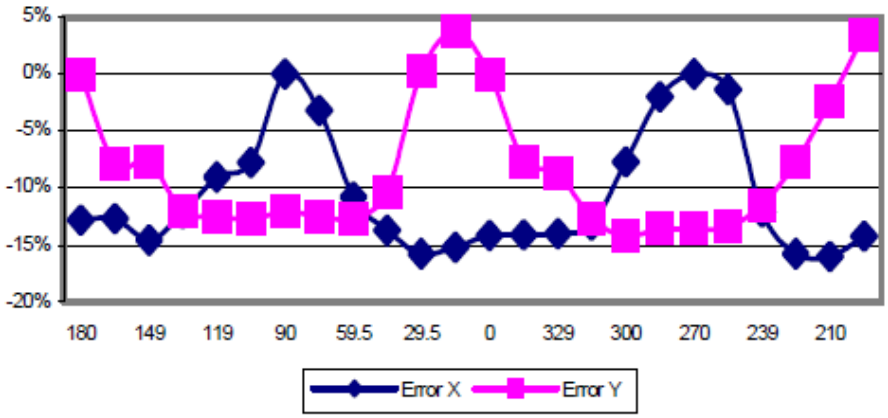


Figura 3.20: Erro na direção X e na direção Y com respeito ao ângulo da câmera com respeito ao eixo Z do alvo.(**fonte:** adaptada de [2])

de que câmeras usualmente tem centro óptico muito próximo centro da matriz de *pixels*.

Tendo em mente essas incertezas quanto a efeitos (supostamente de calibração) e variância de medida tomada em experimento com o robô parado, buscou-se determinar a incerteza associada a equação de medição. Além da variância colhida diretamente do experimento com o robô parado, foi necessário adicionar uma incerteza quanto a localização de cada marcador. Essa incerteza está relacionada à montagem física dos marcadores no teto, que esteve sujeita a erros de medição (foi utilizada uma fita métrica de 20 metros para medir a posição de cada marcador). Buscando cobrir todos os efeitos com certa margem de robustez foi escolhida uma matriz de covariância a estar associada ao ruído de medição:

$$R(k) = \begin{bmatrix} 40000 & 0 & 0 \\ 0 & 40000 & 0 \\ 0 & 0 & 0.0015 \end{bmatrix} \tag{3.12}$$

Em Eq. 3.12 nota-se que os valores admitem uma incerteza de 60 cm ($3\sigma = 600mm$) na medida de x e y recomposta com *ARToolKit*. Embora este valor pareça muito elevado, a utilização do sistema de realidade aumentada para localização encontra justificativa no fato de as medidas fornecidas pelo *ARToolKit* serem absolutas, em contraste as medidas obtidas por odometria que são obtidas por um processo de integração. A integração do erro, ao passar do tempo, leva a resultados extremamente polarizados e não-confiáveis.

Ainda que se considere uma incerteza, o *ARToolKit* é capaz de corrigir a integração de erro feita pela odometria. O fator de inovação gerado a cada medição (medida obtida menos medida esperada) e suposto como uma variável aleatória de média nula. Para eliminar medidas espúrias fornecidas pelo *ARToolKit* (por problemas com a iluminação, por exemplo), foi feito um teste de hipótese, onde se considera que a inovação siga a distribuição χ^2 ate 95%. Medidas fora desta métrica são consideradas dados espúrios, e não geram uma fase de correção do algoritmo do FKU.

Os valores de odometria são relativamente confiáveis para pequenos intervalos de tempo. A modelagem da incerteza do processo de integração das velocidades foi feito considerando o erro do processo como função do erro de velocidade que poderia ser fornecido pelo *encoder*.

$$Q(k) = \begin{bmatrix} v_x^2 \cdot 0.03^2 \cdot 0.1 & 0 & 0 \\ 0 & v_y^2 \cdot 0.03^2 \cdot 0.1 & 0 \\ 0 & 0 & v_\theta^2 \cdot 0.03^2 \cdot 0.003 \end{bmatrix} \tag{3.13}$$

Interpreta-se a Eq. 3.13 da seguinte forma: um erro de velocidade irá compor o erro de localização a cada passo de predição quando integrado durante período de tempo considerado. Com interesse no incremento da matriz de covariâncias (σ^2) toma-se a velocidade em cada direção ao quadrado (com isso elimina-se o efeito do sinal) multiplicado pelo quadrado do tempo de amostragem (idealmente de 0.03 s), e supõe-se que o erro de velocidade seja dado por um percentual da velocidade aplicada (fatores 0.1 e 0.003 na Eq. 3.13).

A seguir, é apresentado um resultado comparativo entre o sistema se localizando apenas por

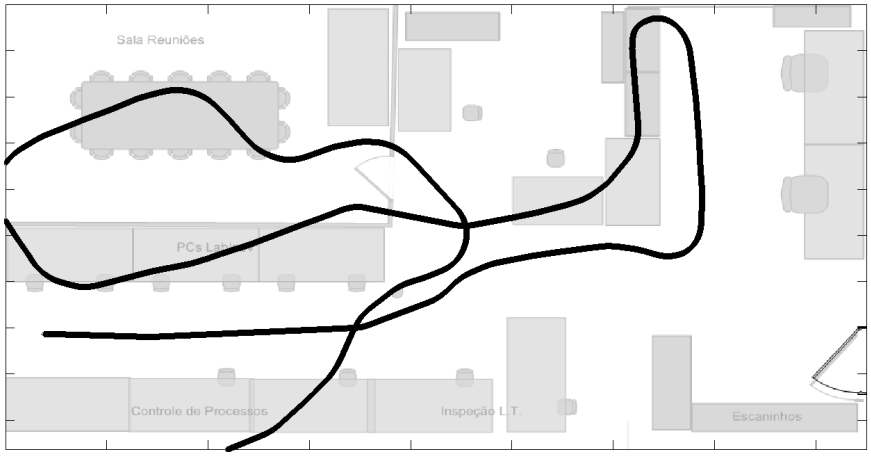


Figura 3.21: Localização do robô numa trajetória utilizando apenas odometria

odometria e, num segundo momento, com o auxílio do *ARToolKit*. A Fig. 3.21 mostra resultados para odometria com correção. Como o tempo de experimento é grande (665 s) a localização não é utilizável. Os círculos em azul são as medidas do sistema de localização com câmeras. Os pontos vermelhos são as medidas descartadas. A trajetória estimada é apresentada em preto.

Na Fig. 3.22 os resultados de estimativa da posição com o filtro em questão são colocados. Os círculos azuis representam medidas de pose fornecidas com o sistema de realidade aumentada. Os círculos azuis que estão marcados em vermelho são aquelas medidas que se realizaram fora do teste de hipótese com χ^2 realizado antes de cada passo de correção.

A Fig. 3.23 traz um aspecto interessante do filtro aplicado para a mesma trajetória da Fig. 3.22: as medidas de inovação geradas em vermelho. Em azul, estão mostrados os limites de 3σ, obtidos da matriz de covariâncias de medida S a cada instante. Quando não há uma medida, considerou-se a inovação nula, pois o aspecto que se quer observar é o espalhamento das inovações com respeito à incerteza da medida. Na verdade, as matrizes Q e R acima foram ajustadas com base nessa dispersão.

Por fim, os resultados da estimação de cada termo do vetor de estados são apresentados na Fig. 3.24

Em detalhe, na Fig. 3.25 é possível perceber o incremento de incerteza para os passos de predição sem presença de medidas e a diminuição quando ocorre o passo de correção.

Ao instante em que mais se acumula incerteza de predição, o intervalo de confiança de 3σ está pouco acima dos 23 cm. Considerando as incertezas associadas ao processo de medição e o fato de, por limitações físicas, haver longos trechos sem marcadores, considera-se o sistema com uso de realidade aumentada com ótimos resultados. O sistema ainda tiraria proveito de uma análise mais delicada quanto a efeitos de calibração e de iluminação. Como é de se esperar, uma calibração menos acurada, ou a incidência não uniforme de luz podem levar a erros polarizados nas medidas.

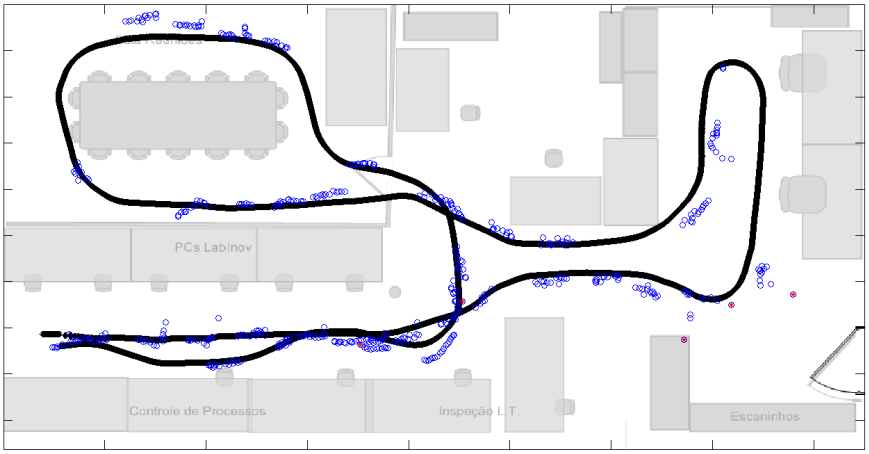


Figura 3.22: Resultado de localização com *ARToolKit* para a mesma trajetória considerada na Fig. 3.21. Pontos azuis representam medidas recebidas. Dentre essas medidas, as descartadas são marcadas em vermelho.

3.5 Desenvolvimento do Controlador de Trajetória de Veículo Auto Guiado

Tendo em mente a fundamentação teórica que foi colocada acerca de controle de trajetórias para veículos com guiagem, buscou-se implementar um controle simples de trajetória com o robô Aramis. Sendo tr um segmento de reta que se deseja seguir, que parte do ponto de coordenadas (x_a, y_a) e segue ao ponto de coordenadas (x_b, y_b), descritas no sistema de coordenadas absoluto (laboratório), com ângulo dado por ϕ_{tr}. Deseja-se caracterizar o erro de desvio da trajetória a cada instante, de forma a determinar uma lei de atuação corrigindo a postura do robô.

O sistema de localização exposto até aqui (Filtro de Kalman *Unscented* utilizando realidade aumentada) forneceu resultados satisfatórios para a realização do controle de trajetória, pois admite apenas variações suaves de posição (condizentes com a cinemática do robô) e combate o efeito de integração do erro presente na localização por odometria.

O desvio instantâneo da trajetória é caracterizado pelo desvio angular e deslocamento linear em relação, sendo o último a distância do centro do robô até a trajetória (err_o) ou a distância do prolongamento da linha de base do robô até a trajetória err_y. Partindo do pressuposto que o sistema de localização fornece a pose do robô $(x_{rob}, y_{rob}, \theta_{rob})$ e notando que $\phi_{tr} = atan2((y_b - y_a), (x_b - x_a))$, é possível descrever o desvio instantâneo de trajetória como:

$$\theta_{erro} = \theta_{\text{rob}} - \phi_{tr} \tag{3.14}$$

$$err_o = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} cos\,(\phi_{tr}) & -sen\,(\phi_{tr}) \\ sen\,(\phi_{tr}) & cos\,(\phi_{tr}) \end{bmatrix} \begin{bmatrix} x_{\text{rob}} - x_a \\ y_{\text{rob}} - y_a \end{bmatrix} \tag{3.15}$$

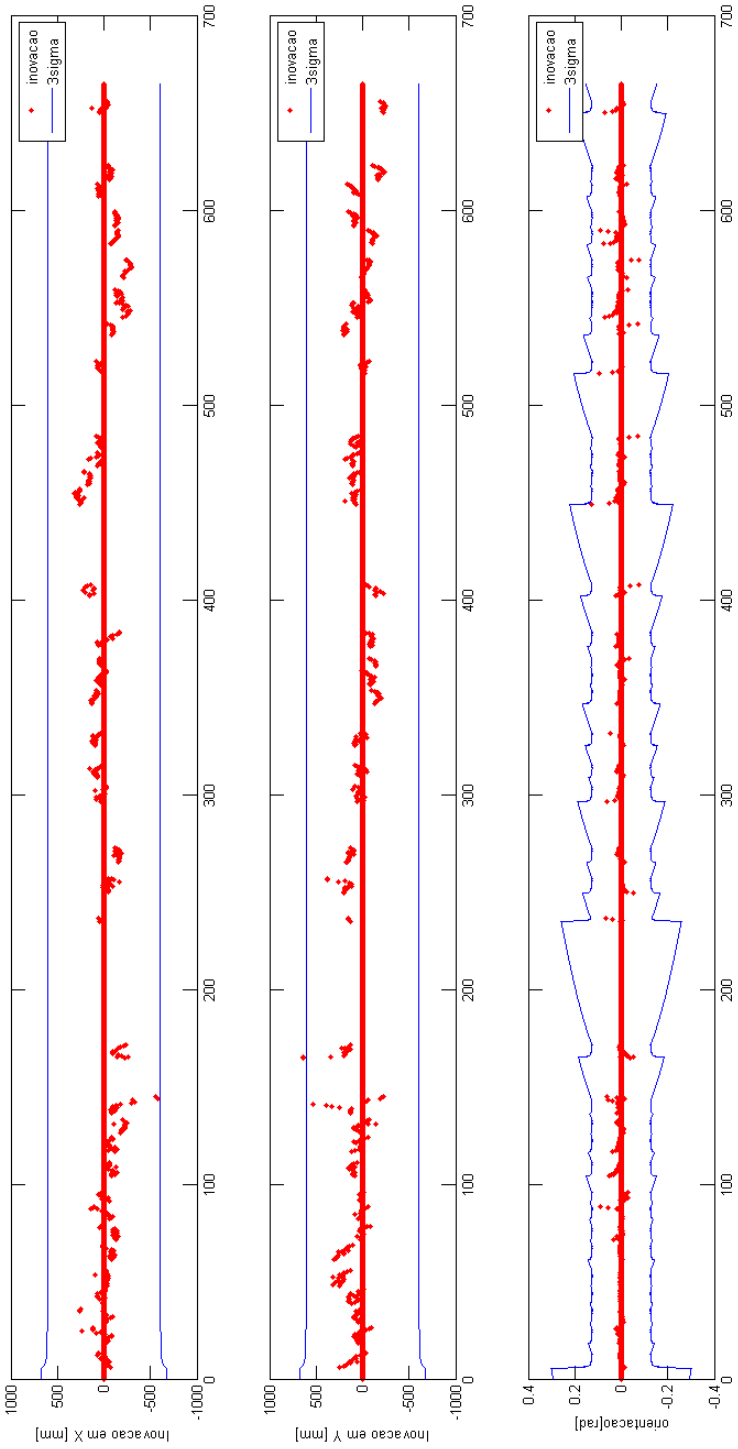


Figura 3.23: Inovações encontradas no filtro de localização para a mesma trajetória considerada na Fig. 3.22

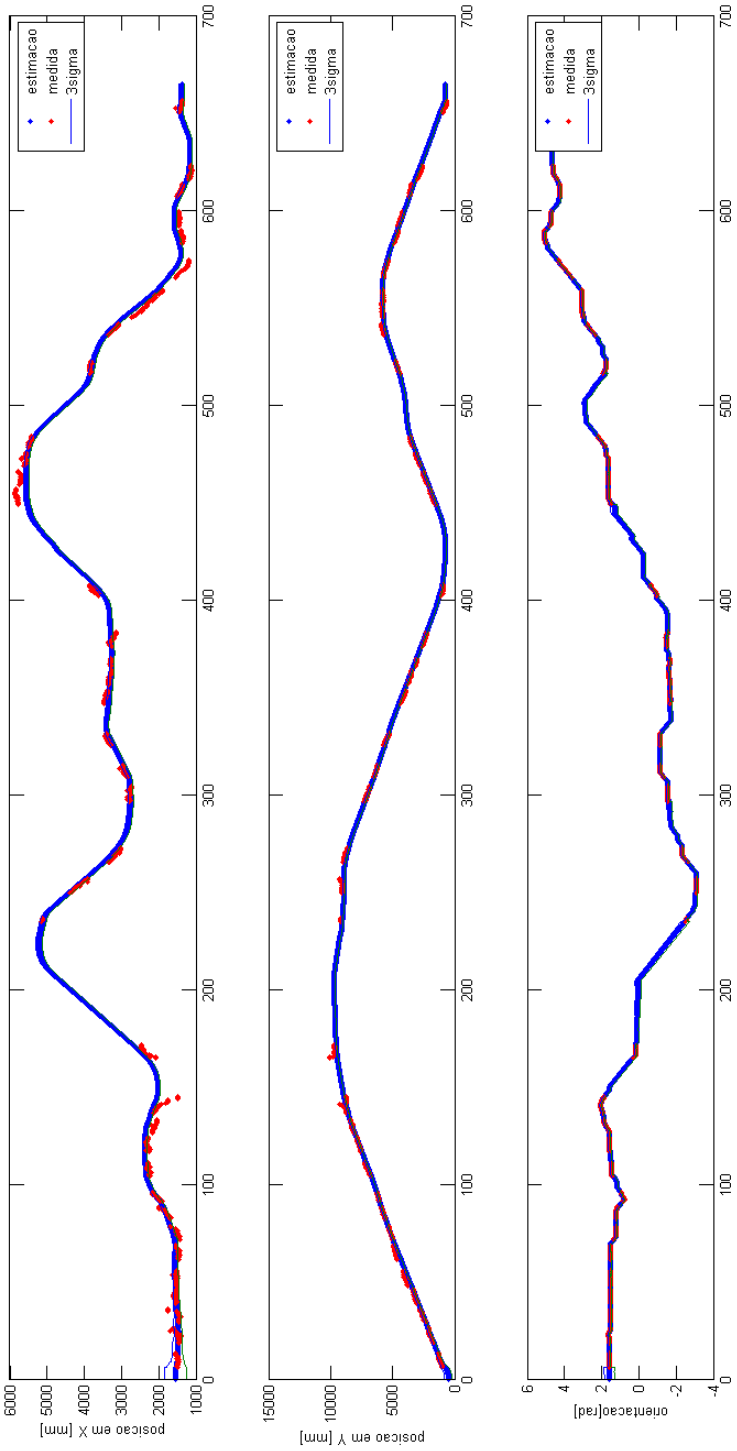


Figura 3.24: Vetor de estados estimado no filtro de localização para a mesma trajetória considerada na Fig. 3.22

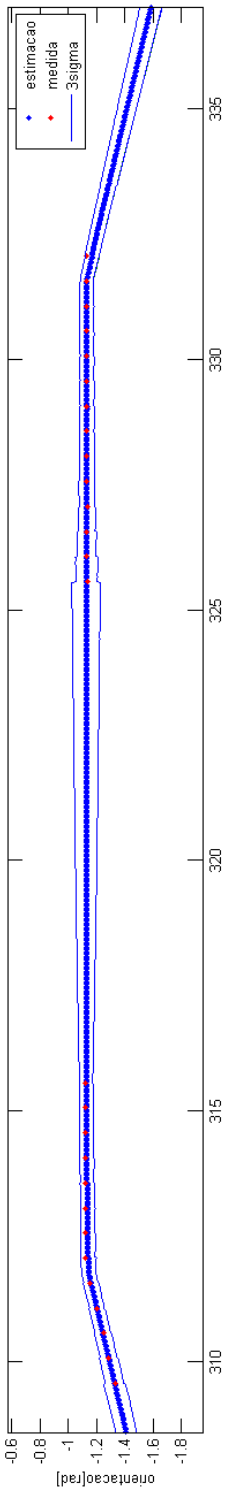


Figura 3.25: Detalhe do gráfico de variável de estado mostrando a incerteza crescente quando na ausência de medidas.

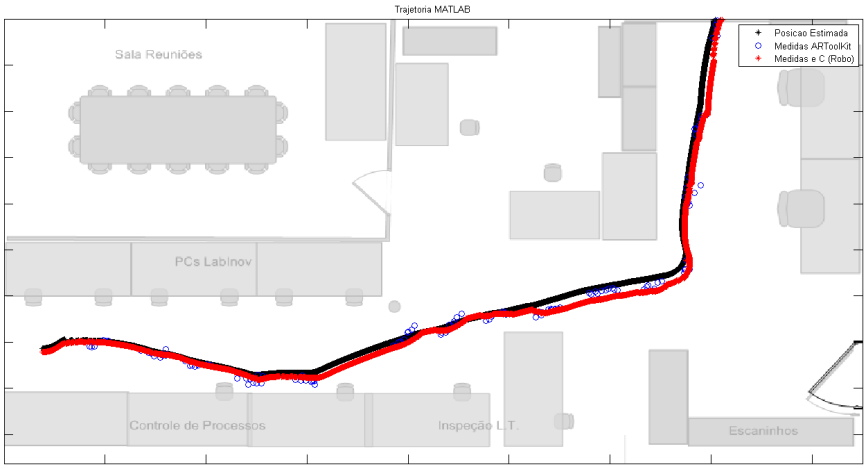
$$err_y = \frac{err_o}{cos\,(\theta_{erro})} \tag{3.16}$$

De posse das equações descritas foi implementado um controle proporcional simples, conforme descrito da fundamentação teórica pela Eq. 2.48. Os ganhos foram inicialmente aproximados pelas equações fornecidas em [21] mas seu ajuste foi feito experimentalmente. A Fig. 3.26 mostra as retas de referência e o resultado para a escolha final de ganhos.

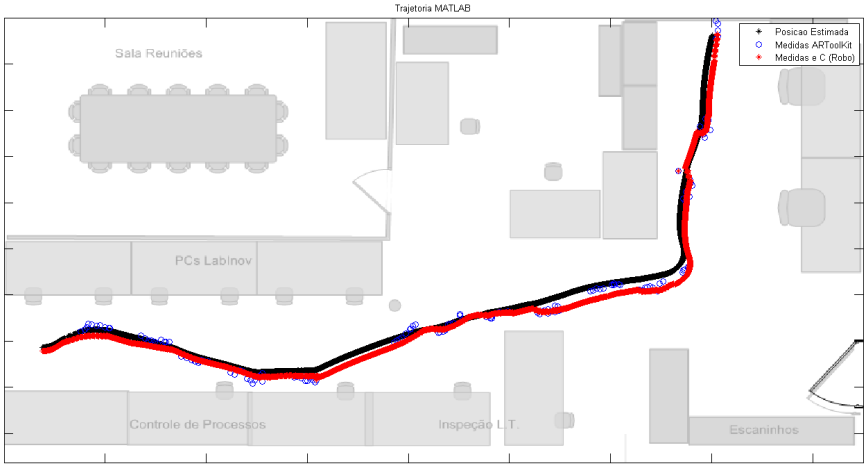
Na Fig. 3.27, é possível ver o resultado do controlador de trajetória para o mesmo experimento repetido três vezes. Em vermelho, é possível ver a posição estimada pelo sistema embarcado no robô, enquanto em preto, é mostrado o resultado estimado *offline* com os dados adquiridos com o *Aramis*. As medidas de *ARToolKit* são mostradas em azul.

Uma análise mais detalhada quanto ao evitamento de obstáculos considerando a faixa de incerteza das medidas, ou um estudo da estabilidade e resposta do sistema, seriam válidos como uma investigação quanto a sistema de navegação para robótica móvel. Todo o substrato para se iniciar algum ensaio nesse sentido está implementado. A comunicação inter-módulos é feita de forma robusta a situações multi-tarefa (*thread-safe*) e os módulos são de fácil compreensão e utilização.

Figura 3.26: Referência de trajetória (linha em verde) e resultado (pontos pretos) obtidos por controle de trajetória.



(a) Experimento 1



(b) Experimento 2

(c) Experimento 3

Figura 3.27: Comparação entre resultados de localização para 3 experimentos seguindo a mesma trajetória.

Capítulo 4

Análise de Modelos Para Sistema de Localização Com RFID

Neste capítulo serão apresentados resultados de testes e investigações com RFID que são relevantes ao desenvolvimento do sistema de localização. É proposta uma forma de resolução do problema e os resultados iniciais são analisados.

4.1 Introdução

Para alcançar o objetivo final de localização utilizando RFID, é inicialmente necessário o desenvolvimento de um trabalho de caracterização do equipamento, levantamento de testes e análise para diferentes ambientes, entre outros aspectos. De posse do sistema auxiliar detalhado no capítulo anterior, o desenvolvimento que será mostrado a seguir ataca o problema de localização com RFID fazendo análises de dados coletados e propondo uma forma de resolução do problema. Os resultados preliminares obtidos são de certa forma animadores, pois apontam uma direção para a resolução do problema.

4.2 Rastreamento por Rádio Frequência

A Identificação por Rádio Frequência, do inglês *Radio Frequency Identification* (RFID), é uma tecnologia de comunicação sem fio a qual permite que computadores leiam e identifiquem *tags* eletrônicas de baixo custo [27]. Essa tecnologia tem seus primeiros relatos na Segunda Guerra Mundial, como forma de identificação de aeronaves amigas. Atualmente, o RFID tem diversas aplicações comerciais, porém, fugiria do escopo do trabalho citá-las, em [28], o autor apresenta algumas delas.

A troca de informações é realizada por ondas eletromagnéticas, tratadas por um circuito RF. É possível que sejam utilizadas diversas faixas de frequência para tal comunicação. A Tabela 4.1 mostra as principais faixas utilizadas para RFID.

Tabela 4.1: Principais faixas de frequência para RFID

Frequência	**Banda**	**RFID típico**
LF	30 - 300 kHz	125 - 134 kHz
HF	3 - 30 MHz	13.56 MHz
UHF	300 MHz - 2 GHz	433 MHz; 956 MHz
Microondas	2 - 30 GHz	2.45 GHz

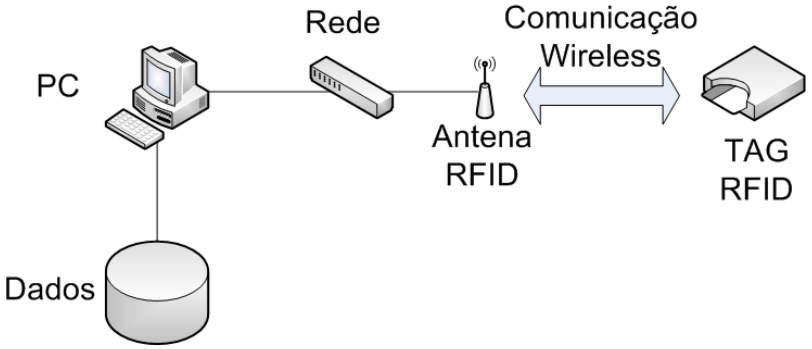


Figura 4.1: Diagrama de sistema utilizando RFID

Os equipamentos típicos desse sistema são os seguintes:

- *Tag*: Dispositivo que transmite informação para a leitora RFID. As *tags* podem ser ativas (alimentadas por bateria interna) ou passivas (circuito excitado por sinal RF externo).
- Antena: Dispositivo que recebe informações enviadas por *tags*.
- Leitora: Dispositivo conectado ao computador, e responsável pelo processamento do sinal, geralmente conectada à antena.

O fluxograma de um sistema que utiliza RFID pode ser visualizado na Fig. 4.1. A *tag* envia um sinal contendo sua identificação, esse sinal será recebido por uma antena. A antena está conectada à leitora, responsável pelo processamento do sinal. É usual ainda que a leitora esteja conectada a um computador para processar as informações das tags, como código, horário de recebimento e potência do sinal. A leitora RFID associa um indicador de força do sinal recebido (RSSI - *Received Signal Strength Indicator*) às suas leituras.

Uma das desvantagens do RFID é a sensibilidade do sinal a interferências externas, o deixando muito ruidoso. Essa falta de robustez pode ser explicada pelo fato de suas aplicações normalmente serem baseadas no reconhecimento de *tags*, ou seja, no simples recebimento do sinal, não em sua análise. Em ambientes convencionais, é normal que o sinal seja atenuado devido a equipamentos eletrônicos, a objetos metálicos e até mesmo a pessoas na proximidade. Um exemplo de interferência em ambientes internos é por múltiplos caminhos, mostrada na Fig. 4.2, quando o receptor recebe o mesmo sinal por diferentes caminhos, dificultando a estimação da distância entre a antena e a *tag*, nesse caso, pode haver interferência construtiva.

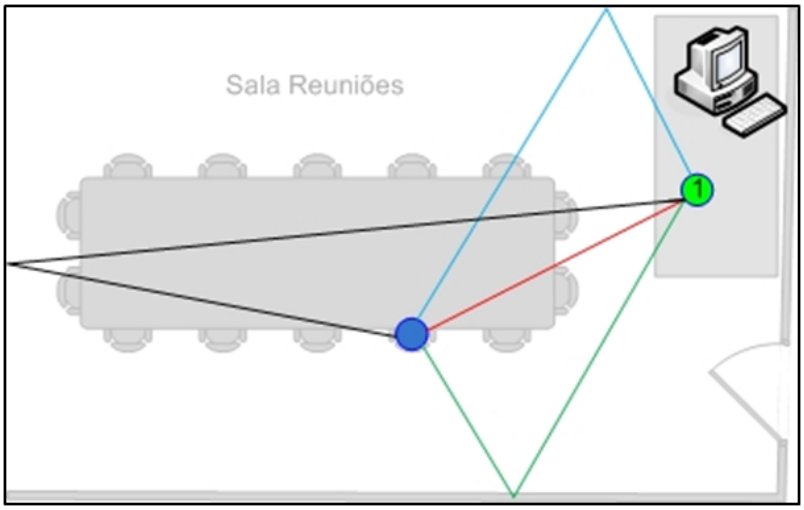


Figura 4.2: Interferência por múltiplos caminhos

(a) Leitora RFID (b) *Tag* RFID

Figura 4.3: Equipamento Wavetrend

4.2.1 Equipamento Utilizado

Para o trabalho, foi utilizado o equipamento disponível no LARA, *tags* ativas e cinco leitoras com antenas, sendo todo o equipamento da marca *Wavetrend*, mostrados na Fig. 4.3. O sistema opera com 433 MHz de frequência, e, segundo o fabricante, tem alcance de $15\,m$. Um sensor de movimento ligado ao circuito interno da *tag* permite regular a taxa de envio de dados para $1.5\,s$ ou $15\,s$, em movimento ou repouso, respectivamente. A comunicação entre os computadores e o equipamento é feita via rede. As leitoras se comportam como servidores, respondendo requisições. As leitoras foram instaladas com endereço fixo na rede do laboratório.

De acordo com o fabricante, há um tratamento interno para a representação do RSSI em um inteiro sem sinal de 8 bits (0 - 255). Também foi afirmado pelo fabricante que, esse sinal era tratado de tal forma que, em um ambiente ideal, sem possibilidade de interferências, a potência varia de acordo com um polinômio do primeiro grau, ou seja, a equação teórica para o equipamento disponível é a mostrada na Eq. 4.1, sendo ρ a distância entre a *tag* e a leitora. Caso o fabricante estivesse certo, seria necessário apenas identificar os parâmetros da Eq. 4.1, mas segundo o próprio fabricante, essas condições de uso são praticamente impossíveis de serem obtidas.

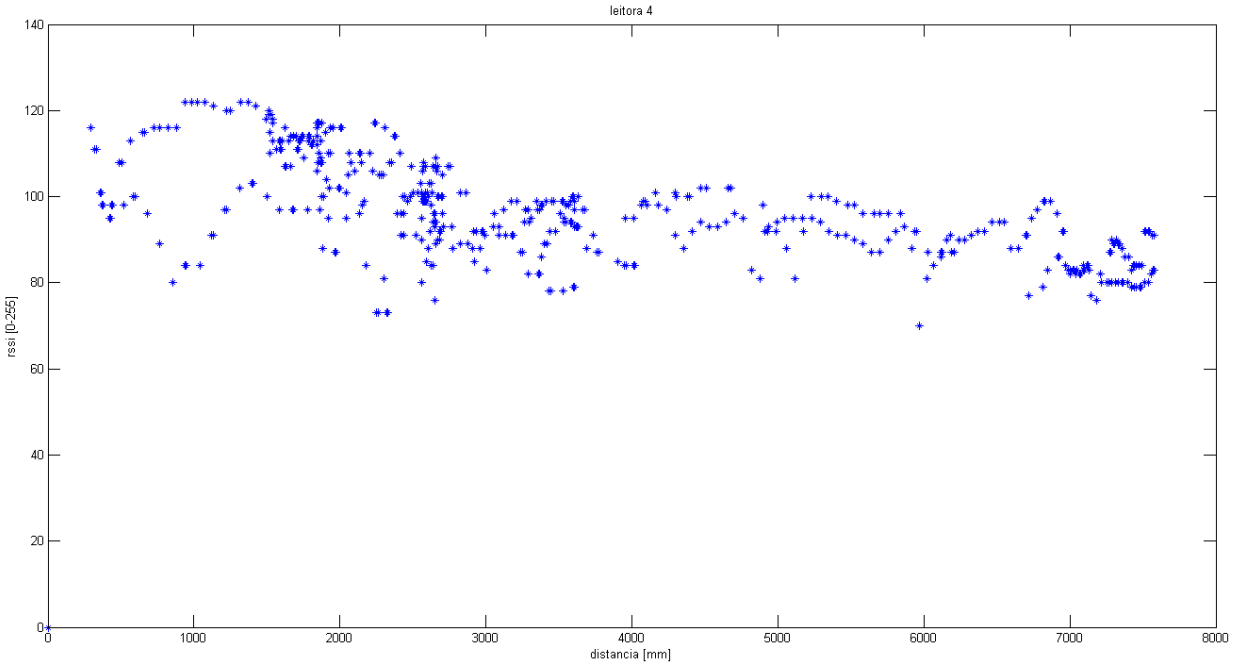


Figura 4.4: Dados experimentais que mostram a variação de RSSI com a distância.

$$\text{RSSI} = K\rho + d \tag{4.1}$$

4.2.2 Experimentos em Ambiente Interno

Foram feitos testes no laboratório para confirmar as informações do fabricante sobre o equipamento. Esses testes de validação foram realizados em conjunto com o trabalho [12], Trabalho de Graduação realizado com o mesmo equipamento, utilizando Redes Neurais Artificiais (RNA) para o propósito de localização utilizando RFID. Os valores apresentados no teste são médias de diversas medições, nunca sendo utilizada apenas uma medição como referência.

O primeiro teste realizado foi de validação da Eq. 4.1. A informação do fabricante não foi confirmada. Foi realizado um teste simples, fazendo medições ao se afastar da leitora por uma reta, medindo distância e RSSI. O resultado obtido é apresentado na Fig. 4.4. Algumas informações importantes são retiradas desse experimento. Fica claro que a curva obtida não é uma reta. Para distâncias acima de 8 m, não foi mais possível ler o sinal das antenas. A informação dada pelo fabricante, fica descartada para ambientes internos como o do laboratório.

O segundo teste realizado mostra que o sinal depende também da taxa de transmissão do sinal RFID. Como mencionado anteriormente, a *tag* tem duas possíveis taxas de transmissão. A Fig. 4.5 mostra um RSSI maior para envios a 1.5 s. Dessa forma, são esperadas relações diferentes entre o sinal e a distância quando o objeto estiver em repouso ou parado.

O terceiro busca verificar a relação entre o sinal recebido e a orientação do transmissor. Ao variar a orientação de uma mesma *tag* à mesma distância, o sinal recebido pela leitora varia.

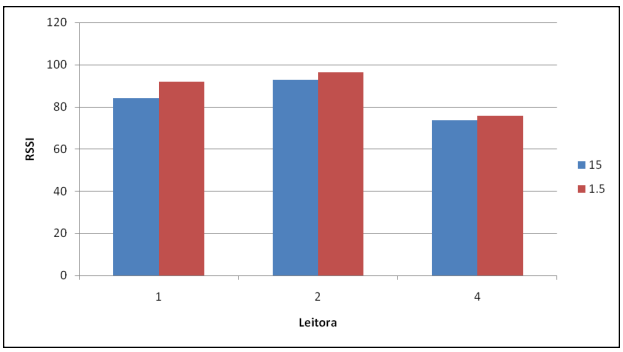


Figura 4.5: Variação de sinal pela taxa de envio

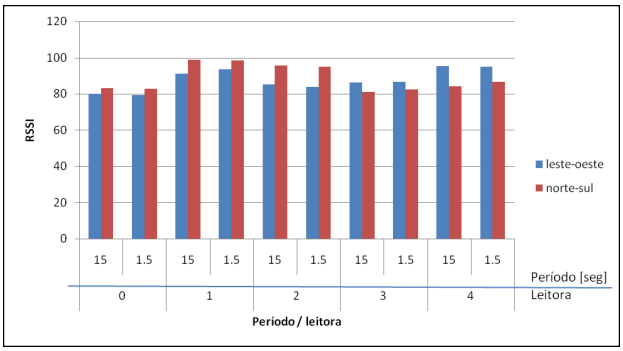


Figura 4.6: Variação de sinal pela orientação

A Fig. 4.6 mostra a diferença entre os sinais a diferentes orientações. Esse resultado pode ser explicado pela antena interna da *tag* não ser omnidirecional, ou seja, seu espectro de irradiação não é uniforme.

Depois, à mesma distância, foram deixadas diferentes *tags* para coleta de dados. Essa aquisição de dados foram feitas uma vez para cada emissor. A Fig. 4.7 mostra a média do sinal recebido de cada uma. É possível notar que o RSSI referente à leitura depende da *tag* utilizada. Esse fenômeno pode ser explicado pela quantidade de transmissões que já foi realizado por cada uma delas, diminuindo a carga de sua bateria ao longo do tempo. No pacote de dados enviado, há um contador que é incrementado a cada transmissão, denominado idade. As idades das *tags* diferem muito entre si. Assim, de acordo que a idade vai aumentando, a potência do sinal enviado diminui.

Por fim, a Fig. 4.8 mostra o RSSI de uma *tag* parada no mesmo local ao longo de um dia. O experimento foi realizado em um fim de semana, dia de baixo movimento no laboratório. É possível ver que o RSSI se manteve constante em horário de menores atividades, como durante a madrugada, por exemplo, comprovando a interferência causada por pessoas no local. Em horários com uma maior movimentação no laboratório, o sinal ficou oscilando bastante, sendo de comportamento difícil de perver.

Em [12], o autor posicionou a mesma *tag* no laboratório em diversos pontos, sempre com a mesma orientação. Deixando-a imóvel durante algum tempo, foram coletados os dados do RSSI recebido por cada leitora, e a média de cada posição foi computada. Com o resultado, o autor treinou uma RNA. A Fig. 4.9 mostra os pontos de coleta de dados em amarelo, e a posição de

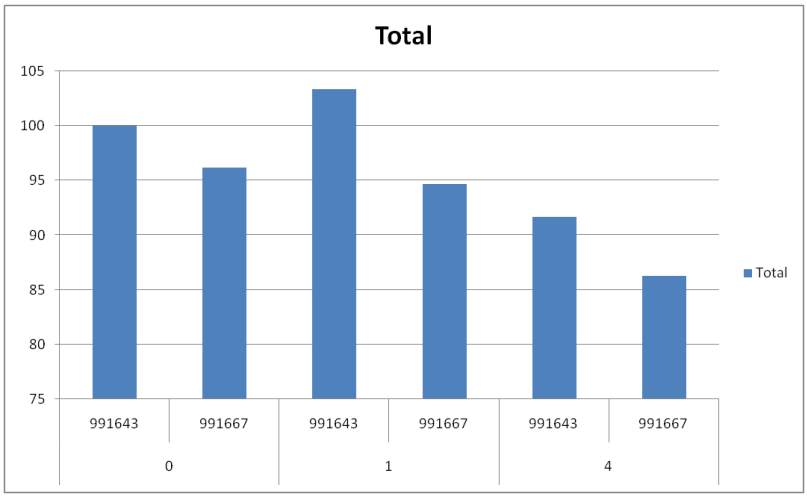


Figura 4.7: Sinal recebido por *tags* diferentes em repouso à mesma distância

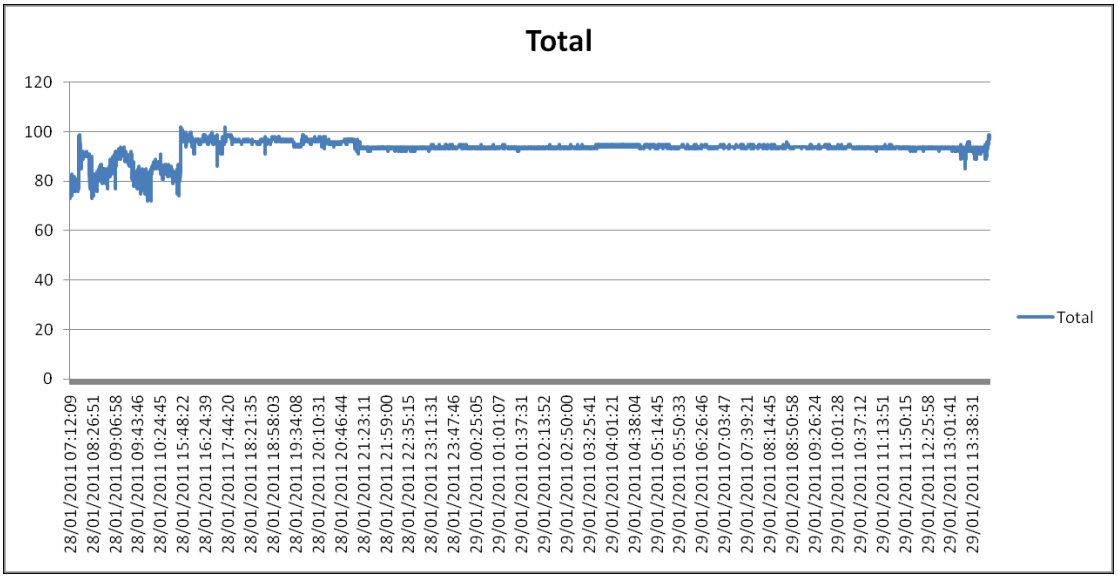


Figura 4.8: Sinal de uma *tag* parada ao longo do dia

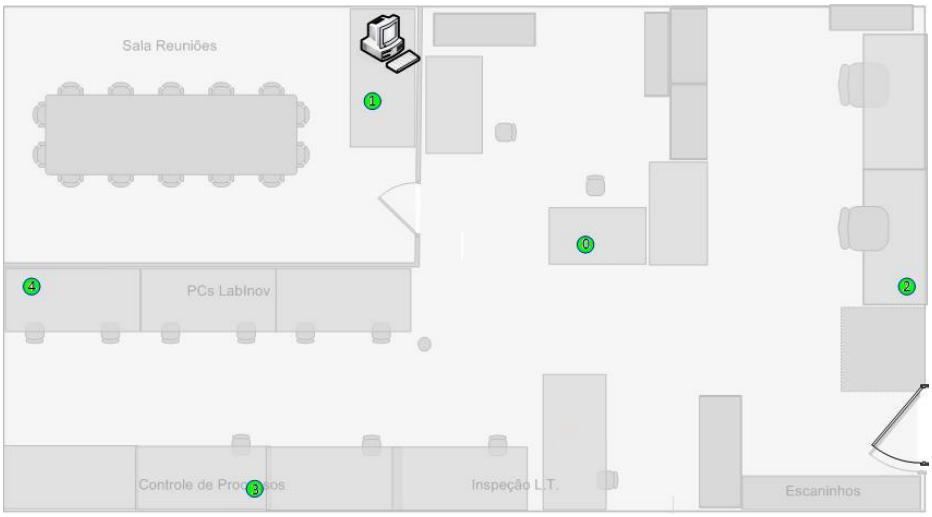


Figura 4.9: Posicionamento das leitoras no laboratório (*números de 0 a 4 correspondem aos IPs 192.168.0.10-14, respectivamente*)

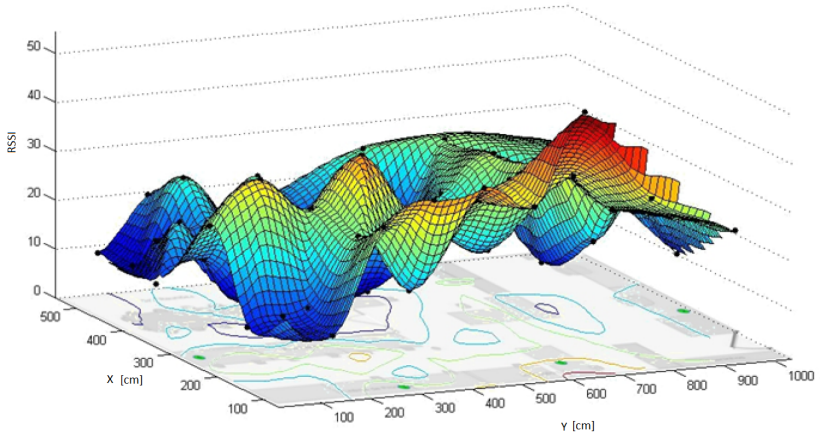


Figura 4.10: Mapeamento do sinal recebido

cada leitora em verde. O mapeamento do RSSI recebido pela segunda leitora é mostrado na Fig. 4.10.

Diante dos dados apresentados, é difícil obter uma relação determinística entre a distância e o RSSI dentro do laboratório. O sistema é influenciado por diversos fatores, inclusive de um modo aparentemente aleatório.

4.2.3 Experimento em Ambiente Externo

Com os experimentos realizados em ambientes internos, não foi possível confirmar a Eq. 4.1. O sinal recebido era muito ruidoso devido ao ambiente do laboratório, com muita interferência eletromagnética e possibilidades de múltiplos caminhos, não sendo possível analisá-lo facilmente. Diante desse cenário, foi realizado um experimento no campo de futebol do Centro Olímpico da UnB. Para que houvesse a menor interferência possível, o teste foi conduzido à noite, horário de

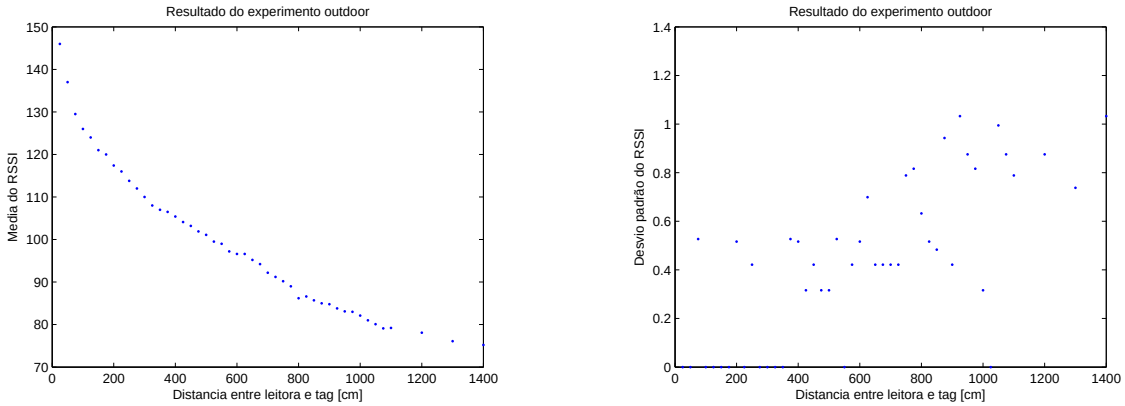
Figura 4.11: Teste em ambiente externo

baixa atividade no local. As pessoas que conduziram os testes ficaram afastadas do local de teste, junto com seus equipamentos (*notebook* e *switch*), se aproximando apenas para reposicionar o experimento. A Fig. 4.11 mostra uma foto da posição do equipamento no campo de futebol, com aproximadamente 15 metros de raio livre.

Foram realizadas dez leituras em cada posição, totalizando 470 amostras, e mantendo sempre a mesma orientação para minimizar sua influência nas medidas. O resultado das medidas pode ser visualizado na Fig. 4.12, a qual mostra o resultado da média e do desvio padrão do RSSI em cada distância. É interessante observar que o desvio padrão é relativamente pequeno quando comparado com as amostras tomadas em ambientes internos, mostrando que os problemas encontrados nestes ambientes são causados por interferência.

Considerando que o RSSI_k dependa apenas da distância ρ_k, foi montada uma matriz de regressores ψ_k, mostrado na Eq. 4.2 para cada uma das N medidas. Analisando a Taxa de Redução de Erro de cada regressor, foi possível determinar quais seriam os parâmetros mais influente para um modelo polinomial. O resultado do algoritmo é mostrado na Tab. 4.2. A Fig. reffig:ERR mostra o resultado para o algoritmo de Taxa de Redução de Erro, com uma ampliação para que os resultados obtidos fiquem claros.

$$\psi_k = \begin{bmatrix} u(1)^0 & u(1)^1 & u(1)^2 & \cdots & u(1)^{10} \\ u(2)^0 & u(2)^1 & u(2)^2 & \cdots & u(2)^{10} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u(N)^0 & u(N)^1 & u(N)^2 & \cdots & u(N)^{10} \end{bmatrix} \tag{4.2}$$



(a) Média do RSSI

(b) Desvio padrão do RSSI

Figura 4.12: Resultado medição em ambiente externo

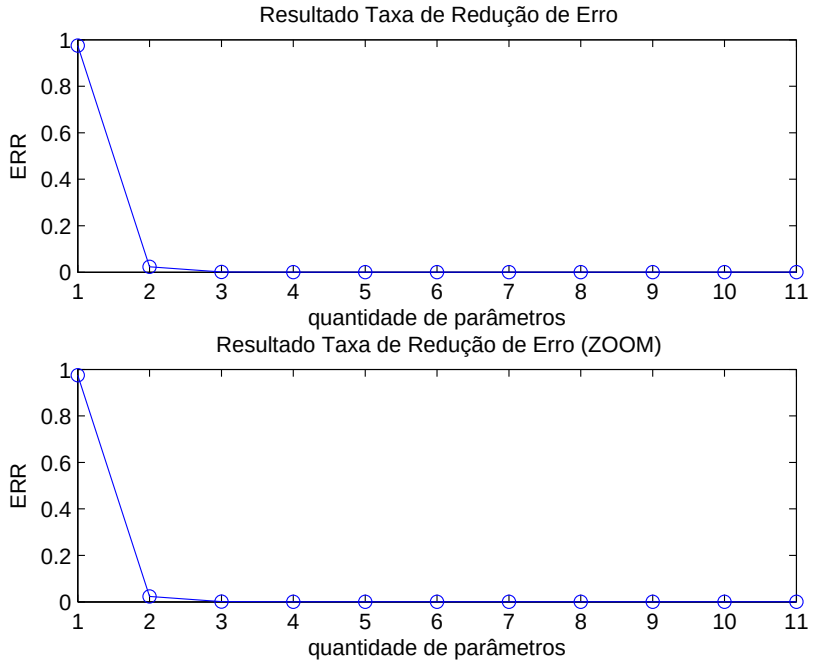


Figura 4.13: Resultado da taxa de redução de erro

Tabela 4.2: Resultado da taxa de redução de erro

Regressor	**ERR**
constante	0.9751
u_k	0.0234
u_k^2	0.0013
u_k^3	$4.943 \cdots 10^{-5}$
u_k^4	$8.594 \cdot 10^{-5}$
u_k^5	$4.644 \cdots 10^{-5}$
u_k^6	$2.759 \cdots 10^{-6}$
u_k^7	$5.133 \cdots 10^{-9}$
u_k^8	$5.267 \cdots 10^{-6}$
u_k^9	$1.305 \cdots 10^{-5}$
u_k^{10}	$6.561 \cdots 10^{-6}$

Analisando a Tab. 4.2, foi decidido utilizar um polinômio de segundo grau para representar a relação entre o RSSI e a distância entre a *tag* e a leitora. Com esse polinômio, é possível reduzir consideravelmente o erro, segundo os dados colhidos, e, por se tratarem apenas de três parâmetros, não existe um esforço computacional muito grande. Para a identificação dos parâmetros, foi utilizado o Método dos Mínimos Quadrados (MMQ), sendo o modelo encontrado mostrado na Eq. 4.3, com um erro médio quadrático de 1.563. O ajuste da curva aos pontos pode ser visualizada na Fig. 4.14. É interessante observar que, caso não tivesse sido utilizado a Taxa de Redução de Erro, provavelmente o parâmetro quadrático seria descartado, por ser de uma ordem de grande muito menor do que os outros, ele diminui bastante o erro, sendo significativo.

$$\text{RSSI} = 2.7678 \cdot 10^{-5}\rho^2 - 0.0798\rho + 133.6599 \tag{4.3}$$

Considerando que cada par *tag* - antena tenha relação particular entre RSSI e distância, é razoável que a estrutura utilizada para a estimação da distância a partir do RSSI seja como a mostrada na Eq. 4.3, porém, com outros parâmetros.

4.3 Sistema de Localização com RFID

Pelo que foi observado, há uma grande dificuldade em encontrar uma função que relacione RSSI e distância para o caso ambientes internos. Diversos efeitos podem alterar a potência do sinal recebido em ambientes gerais (como num laboratório de robótica por exemplo), tais como multicaminhos, reflexões, refrações, atenuações, interferências ou alterações abruptas e imprevisíveis do ambiente. Na verdade, é intuitivo que a potência do sinal recebido não seja a mesma para diferentes pontos situados a uma mesma distância do receptor, dado a presença de obstáculos que podem afetar diferentemente cada posição.

Um sistema de localização baseado em RFID deveria modelar a relação entre potência do

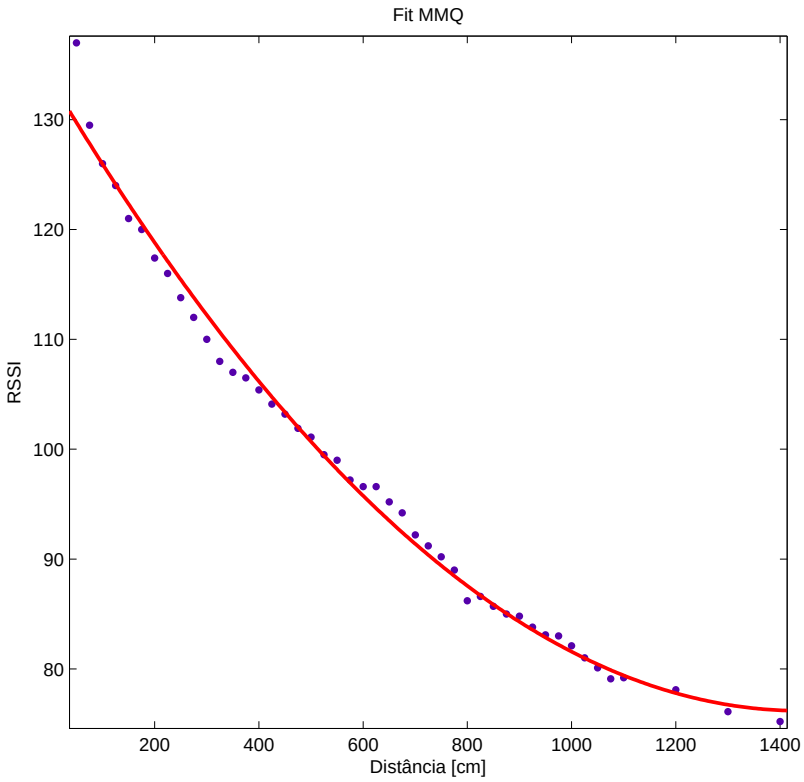


Figura 4.14: Resultado mínimos quadrados

sinal recebido e distância entre emissor e receptor (a princípio, seria muito interessante se obter uma relação entre potência do sinal recebido e posição relativa da *tag*, mas se sabe previamente, da teoria de propagação de ondas, que a potência deve variar conforme a distância). Se fosse possível descrever uma modelagem parametrizada contabilizando os efeitos que afetam diretamente o sinal para ambientes internos, alcançaría-se um modelo de alta complexidade, que deveria levar em consideração fatores para os quais não se tem uma mensuração direta, como presença de pessoas, temperatura ou alterações no ambiente. Diante de problemas de alta complexidade, muitas vezes se fazem válidas soluções relativamente simples. Buscou-se uma forma de inserir ao modelo características que se conheciam do sistema, e encapsular efeitos desconhecidos de forma coerente no modelo. Foi considerado que o sinal mais puro (livre de interferências) que o receptor poderia receber num ambiente interno como o do laboratório, refletiria numa medida de potência condizente com as recebidas no teste de ambientes externos realizado no Centro Olímpico da Universidade. Contudo, o sinal geralmente chega atenuado ao receptor tendo sofrido algum efeito inerente ao ambiente desfavorável do laboratório e apresenta uma medida RSSI inferior ao esperado com base no ensaio externo. Assim, foi sugerido que a relação no ambiente interno seja dada como na Eq. 4.4, que é a relação em ambiente ideal multiplicada por um fator. O termo $(1 - \alpha)$ representa uma possível atenuação que ocorre com o sinal. Esperava-se inicialmente que $0 \leq \alpha \leq 1$, havendo apenas efeitos atenuadores. Contudo, em alguns casos, α assume valores pouco menores que 0, o que pode representar uma interferência construtiva no sinal, algum efeito espúrio

na eletrônica de emissão, ou mesmo a diferença de sinal recebido dado por diferentes leitoras.

$$\text{RSSI} = \left(2.7678 \cdot 10^{-5}\rho^2 - 0.0798\rho + 133.6599\right)(1-\alpha) \tag{4.4}$$

É esperado que o fator α varie suavemente em posições adjacentes. Essa abordagem trás pouco equacionamento quanto a efeitos de ambientes internos, encapsulando-os num fator de atenuação. Contudo, é mais interessante do que uma abordagem clássica que modele divergências entre valor esperado e valor medido por um erro aleatório, deixando a cargo do erro todos os fatores desconhecidos. Tal abordagem estenderia o vetor de estados pelos parâmetros de cálculo do valor de RSSI (3 por leitora) e guardaria menos relação com o princípio físico de propagação. Com a utilização do modelo proposto, foi possível obter resultados interessantes com um vetor de estados com 7 elementos, como será mostrado adiante.

Foi utilizado o FKU para estimar α_i em cada posição (5 últimos termos do vetor de estados na Eq. 4.5), sendo $0 \leq i \leq 4$, indicando o índice da leitora RFID. Além desse parâmetro, é feito um ajuste fino de alguns parâmetros obtidos no experimento de ambiente externo (primeiros 2 termos na Eq. 4.5). O termo constante na equação 4.3 não é estimado, pois ele representa um limite do sinal recebido, e foi considerado fixo, retirando uma variável de estado do filtro. O modelo de predição utilizado para o processo é apresentado na Eq. 4.5.

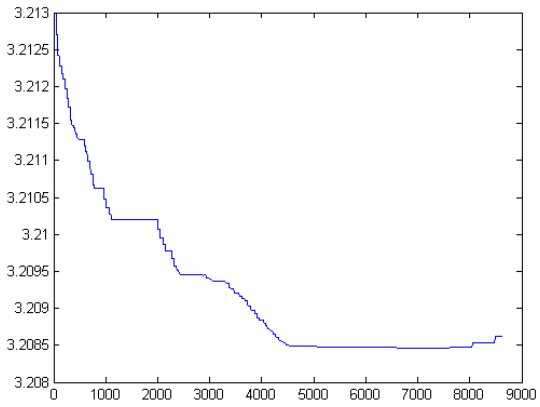
$$\begin{bmatrix} a_{k+1} \\ b_{k+1} \\ \alpha_{0,k+1} \\ \alpha_{1,k+1} \\ \alpha_{2,k+1} \\ \alpha_{3,k+1} \\ \alpha_{4,k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_k \\ b_k \\ \alpha_{0,k} \\ \alpha_{1,k} \\ \alpha_{2,k} \\ \alpha_{3,k} \\ \alpha_{4,k} \end{bmatrix} \tag{4.5}$$

A não linearidade do modelo para justificar o uso do FKU na verdade se encontra na parte da medição. A Eq. 4.6 mostra o modelo de medição. A leitora i está no ponto (x_{ri}, y_{ri}), e a estimativa de posição do robô (x_k, y_k) é fornecidas por medidas feitas com o *ARToolKit*.

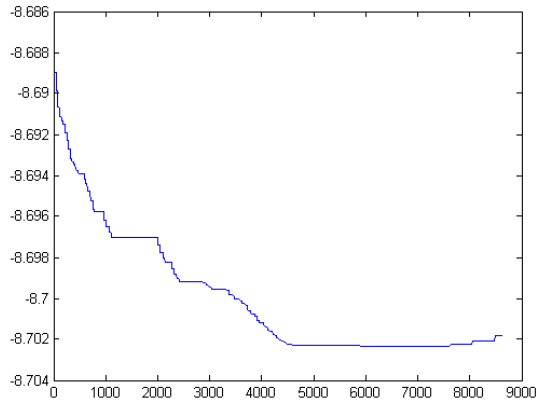
$$\begin{aligned} \rho_{i,k} &= \sqrt{(x_{ri} - x_k)^2 + (y_{ri} - y_k)^2} \\ \text{RSSI}_{i,k} &= \left(a_k\rho_{i,k}^2 + b_k\rho_{i,k} + 133.6599\right)(1 - \alpha_{i,k}) \end{aligned} \tag{4.6}$$

Um teste foi realizado deixando o robô imóvel para verificar a convergência dos parâmetros conforme a modelagem de incerteza a ser dada a cada parâmetro (maior incerteza para atenuação e menor incerteza para os parâmetros obtidos em ambiente externo). A Fig. 4.15 apresenta a estimação dos parâmetros do modelo que fazem o cálculo do sinal esperado pelo experimento em ambiente externo. É visível que houve pouca variação, realmente se tratando de um ajuste fino.

Os valores de α variaram rapidamente no início, tendendo a se estabilizar, como mostra a Fig. 4.16.

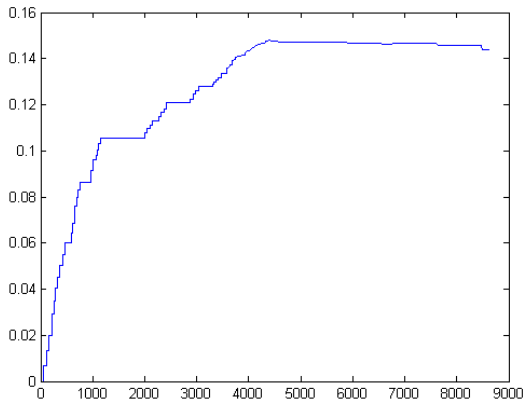


(a) Parâmetro do termo de segundo grau

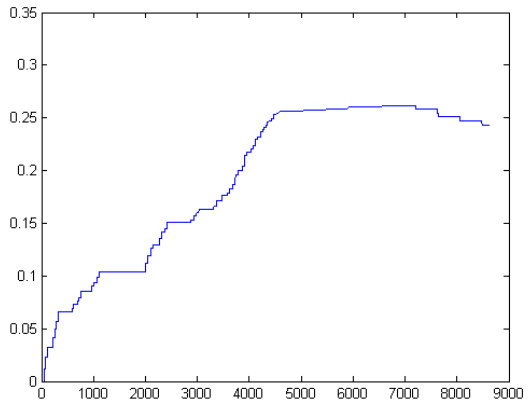


(b) Parâmetro do termo de primeiro grau

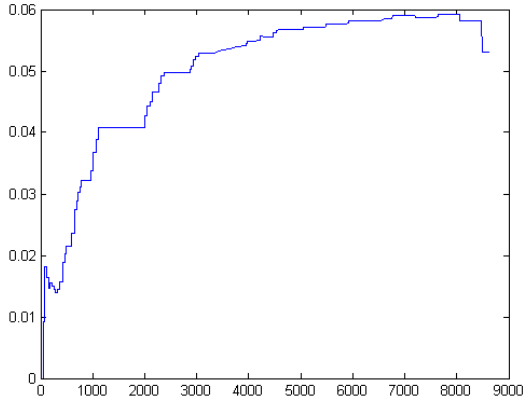
Figura 4.15: Ajuste de termos do modelo para ambientes externos



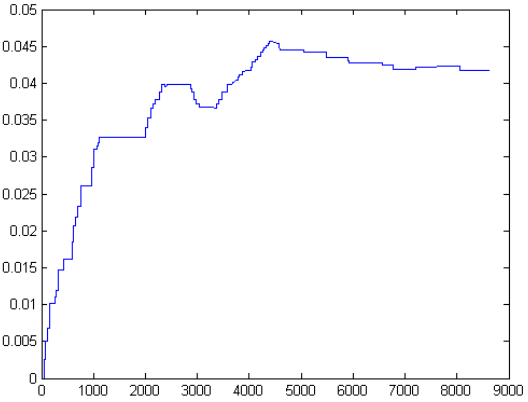
(a) α_0



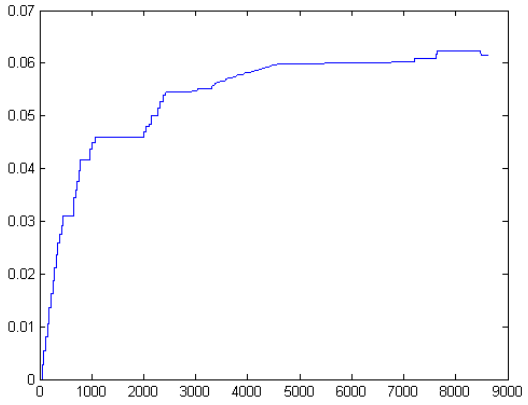
(b) α_1



(c) α_2



(d) α_3



(e) α_4

Figura 4.16: Ajuste de α para cada leitora pelo número de iterações

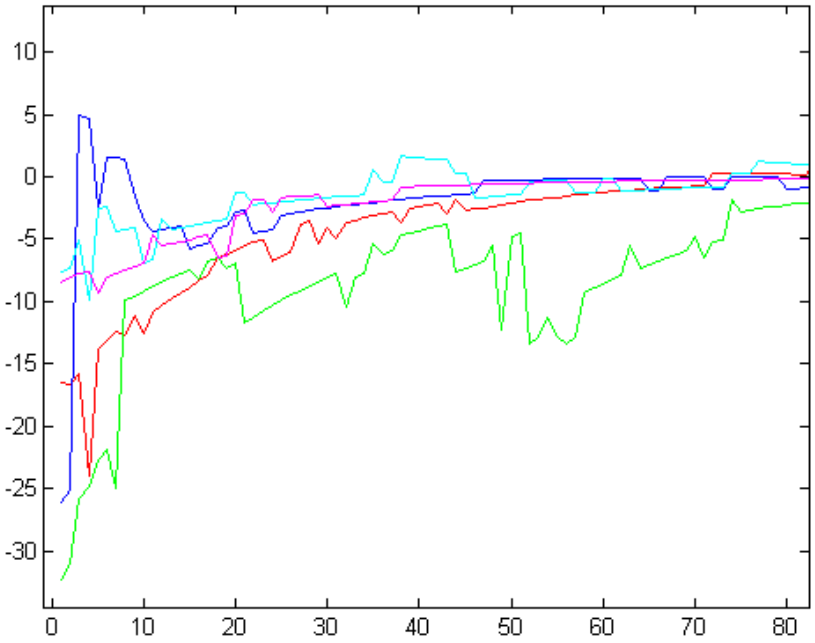


Figura 4.17: Com o passar das iterações num teste com o robô parado (como o robô esteve parado, os índices no eixo X evoluem apenas a cada 15 segundo) a inovação para cada leitora (diferentes cores) tende a zero

A Fig. 4.17 mostra a inovação do sistema a cada instante. Nota-se que houve uma tendência a diminuir, mesmo com o sinal ruidoso, refletindo que a fase de predição tinha uma boa estiativa do resultado que deveria ser recebido.

De posse desses resultados, foi imediata a proposta de um filtro que tivesse como vetor de estados a pose do robô no laboratório e os parâmetros α para cada leitora, compondo um vetor de tamanho $n = 8$. Na primeira modelagem deste filtro, percebeu-se que faltava um equacionamento para as incertezas de modo que o filtro corrigisse posição e parâmetros de atenuação de maneira condizente com a realidade. Ou seja, em algumas situações, uma variação provocada propositalmente na posição do robô no mundo real gerava um fator de inovação nas medidas de RSSI que, embora se esperasse que corrigisse a posição, alterava muito mais os valores dos parâmetros de atenuação, de forma a explicar os sinais recebidos. Por outro lado, aumentando a certeza dos parâmetros de atenuação e diminuindo a certeza com respeito à posição do robô, o sistema de localização é penalizado nos casos em que o sinal é atenuado por uma alteração no ambiente sem que a posição mude, pois o sistema utilizará o fator de inovação gerado pela alteração no ambiente para corrigir a posição do robô, uma vez que esta tem baixa confiança.

Diante da dificuldade de modelar a incerteza relativa entre posição e atenuação para se obter melhores resultados, ficou clara a ideia de que uma informação prévia com respeito a atenuação de cada ponto seria de grande valia para a localização. Diversas técnicas poderiam ser implementadas para se alcançar essa condição. Entre elas, poderia ser feita uma "calibração" do ambiente

tomando valores de RSSI em posições conhecidas e levantando um mapa de atenuação do ambiente. Essa calibração, numa abordagem mais simplória, seria realizada uma única vez guardando seus resultados para uso posterior e teria a restrição de um ambiente que não sofresse alterações significativas. Num cenário de configuração física volátil, como o caso de interesse (LARA), o mapa de atenuações uma vez levantado deveria ter um algoritmo de atualização, pois se a cada iteração se colhe informação do mapa e o realimenta, se confere maior plasticidade ao sistema. Por fim, a utilização de *tags* fixas em posições conhecidas pode oferecer ao sistema o valor de atenuação para determinado ponto a cada instante, mantendo um mapa de atenuação sempre calibrado.

De forma a obter a informação prévia de atenuação foi proposto um modelo fazendo uso da ferramenta disponível implementada na plataforma Aramis: o sistema de localização com *ARToolKit*. Um filtro como o descrito pelas Eq. 4.5 e 4.6 estimava a cada instante um valor de α para cada leitora, e este valor em seguida era utilizado para estimar a posição do robô por um segundo filtro, descrito pelas equações Eq. 4.7 e 4.8. Na fase de medição, a estimativa de posição (x_k, y_k) é retirada d vetor de estados, e o valor do parâmetro de atenuação α é fornecido pelo filtro das equações Eq. 4.5 e 4.6.

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \theta_{k+1} \\ \alpha_{0,k+1} \\ \alpha_{1,k+1} \\ \alpha_{2,k+1} \\ \alpha_{3,k+1} \\ \alpha_{4,k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ \alpha_{0,k} \\ \alpha_{1,k} \\ \alpha_{2,k} \\ \alpha_{3,k} \\ \alpha_{4,k} \end{bmatrix} \begin{bmatrix} cos\,(\theta) & -sen\,(\theta) & 0 \\ sen\,(\theta) & cos\,(\theta) & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{v_r+v_l}{2} \\ 0 \\ \frac{-v_r+v_l}{2l} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot dt \tag{4.7}$$

$$\begin{aligned} \rho_{i,k} &= \sqrt{(x_{ri} - x_k)^2 + (y_{ri} - y_k)^2} \\ \mathrm{RSSI}_{i,k} &= \left(a_k \rho_{i,k}^2 + b_k \rho_{i,k} + 133.6599\right)(1 - \alpha_{i,k}) \end{aligned} \tag{4.8}$$

Na Eq. 4.8 (x_{ri}, y_{ri}) são as coordenadas da leitora i no laboratório.

Cabe um comentário sobre a proposta de localização formulada. A princípio fica uma sensação de redundância no algoritmo, pois foi utilizada uma estimativa prévia de localização num primeiro momento, para gerar os parâmetros de atenuação e só depois utiliza-se o filtro com valores de RSSI para determinar uma estimativa da posição do robô. De fato, esse caráter implica que está sendo feita uma forma de fusão sensorial entre RFID e *ARToolKit*. Porém, a implementação e documentação dos resultados finais é válida, pois na fase do filtro em que se faz a estimativa com RFID, já não se utiliza mais informações do *ARToolKit*, exceto pelo fator de atenuação que foi gerado a partir de seus dados. Em outras palavras, a interface entre os dois filtros se dá no fornecimento de uma forma de relacionar RSSI à distância entre leitora e emissor RFID. Bons resultados com esse sistema deixam claro que a utilização de um técnica para se determinar a atenuação do sinal (por exemplo, alguma dentre as citadas anteriormente) ou mesmo uma nova proposta de modelagem da relação RSSI por distância poderia alcançar resultados razoáveis de

localização utilizando o RFID.

Para avaliar os resultados dessa proposta foi tomado um conjunto de dados de avaliação de seguimento de uma trajetória como a mostrada na Fig. 3.26. O resultado da odometria sem correção é apresentado na Fig. 4.18. É possível ver que, se esse sistema fosse utilizado para navegação ou mesmo para montar o mapeamento de α, o resultado seria completamente incoerente, visto que a localização passa por locais absurdos, como por cima dos móveis do laboratório.

Figura 4.18: Resultado de localização para os dados colhidos no experimento citado anteriormente e ilustrado na Fig. 3.26, utilizando apenas odometria

A Fig. 4.19 apresenta o resultado da trajetória realizada pelo robô. Esse resultado, já corrigido pelo sistema de visão computacional, foi utilizado determinar os parâmetros α para cada leitora, permitindo que fosse feita uma análise do sinal. Os valores de α foram tomados com um ajuste de forma a confiar mais no modelo de predição do que no modelo de medição, isso confere uma característica de suavidade nas variações de α, por mais ruidoso que seja o sinal de RSSI. A interpolação suave dos valores de atenuação entre posições é um fator interessante, pois favorece a utilização de técnicas como a adoção de *tags* fixas, onde dispondo da atenuação de alguns pontos interpola-se a atenuação de pontos entre eles.

A Fig. 4.20 mostra o resultado da estimativa de α's durante o percurso. A variação foi suave, como era esperado, pois essa variação foi modelada variando lentamente com a variação de posição (o modelo de evolução utilizado, com alta confiança, limita a variação dos parâmetros a cada instante).

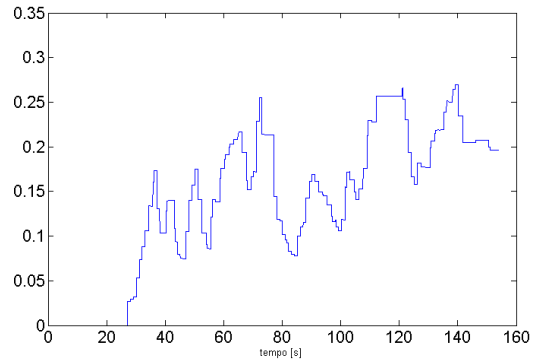
Utilizando α_i como entrada do sistema, é possível estimar a posição do robô e as medidas de RSSI. Um comparativo entre os trajetos estimados é apresentado na Fig. 4.21. Na Fig. 4.21 o trajeto em vermelho representa a localização dada por odometria. Em preto observa-se a curva estimada pelo sistema de localização com *ARToolKit* e em verde é a estimativa utilizando o RFID. Percebe-se que a correção por RFID melhora os resultado da odometria, mas o modelo ainda não é tão bom, visto que apresenta resultados inferiores a localização com visão computacional. Na trajetória estimada pelo RFID (em verde), nota-se que o robô se afasta da trajetória de uma forma

Figura 4.19: Resultado de localização para os dados colhidos no experimento citado anteriormente e ilustrado na Fig. 3.26, utilizando sistema de localização com *ARToolKit*

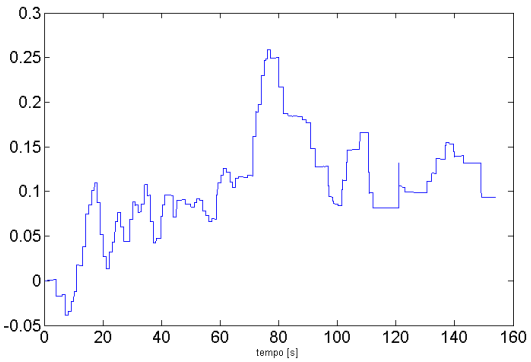
que não foi observada na realização do experimento quando há uma curva mais acentuada. Esse efeito pode estar relacionado a uma deficiência de modelo, pois não é feita nenhuma consideração sobre o angulo relativo entre *tag* e leitora para estimação do RSSI, enquanto que no experimento realizado em ambiente externo, ficou claro que a emissão não era exatamente igual em todas as direções. Assim uma variação da orientação gerada por uma curva, pode gerar uma variação dos sinais RSSI que seria traduzida em deslocamento pelo modelo.

Para verificar se a variação de α_i é coerente, a Fig. 4.22 mostra, para uma leitora, como varia α_i em duas trajetórias controladas. Essa análise é complicada devido à sincronização. A métrica de comparar por tempo de cada valor de atenuação calculado seria incoerente, pois os relógios nos dois experimentos estão dessincronizados. A ideal seria comparar a atenuação por posição ao longo da trajetória. Como se percebe no gráfico da trajetória, a posição é função da coordenada y por causa da escolha do traço da trajetória. Assim foi plotado o valor de atenuação para suas trajetorias controladas conforma são mostrados em Fig. 4.22.

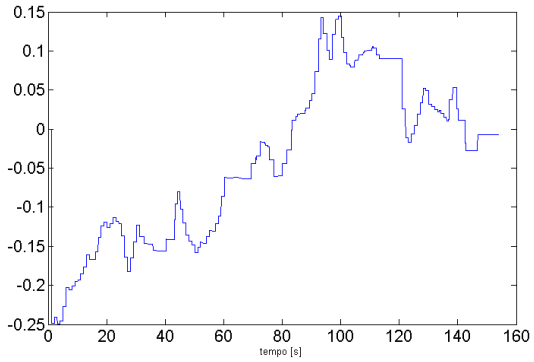
A análise da imagem não é trivial pois percebe-se a Por uma análise inicial da imagem, percebe-se que a variação é semelhante apresenta alguma semelhança. A nuvem de pontos parece se mover de maneira similar ao longo da trajetória, o que mostraria ser coerente a abordagem de um mapa de atenuação com algum mecanismo de atualização para se readaptar a alterações no ambiente. Uma dificuldade encontrada para realizar tal análise é a alta variância do sinal recebido.



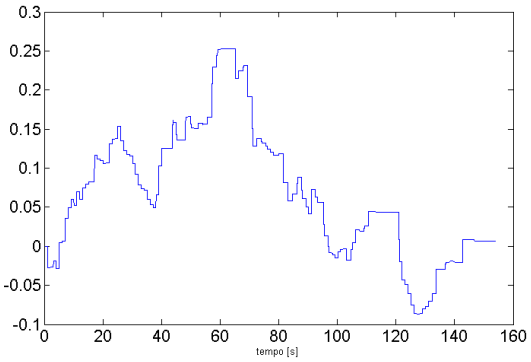
(a) α_0



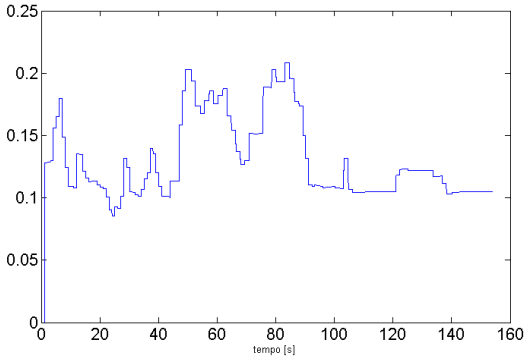
(b) α_1



(c) α_2



(d) α_3



(e) α_4

Figura 4.20: Ajuste de α para cada leitora com robô em movimento, visto em função das iterações (como o robô esteve em movimento, os índices no eixo X evoluem a cada 1.5 segundos)

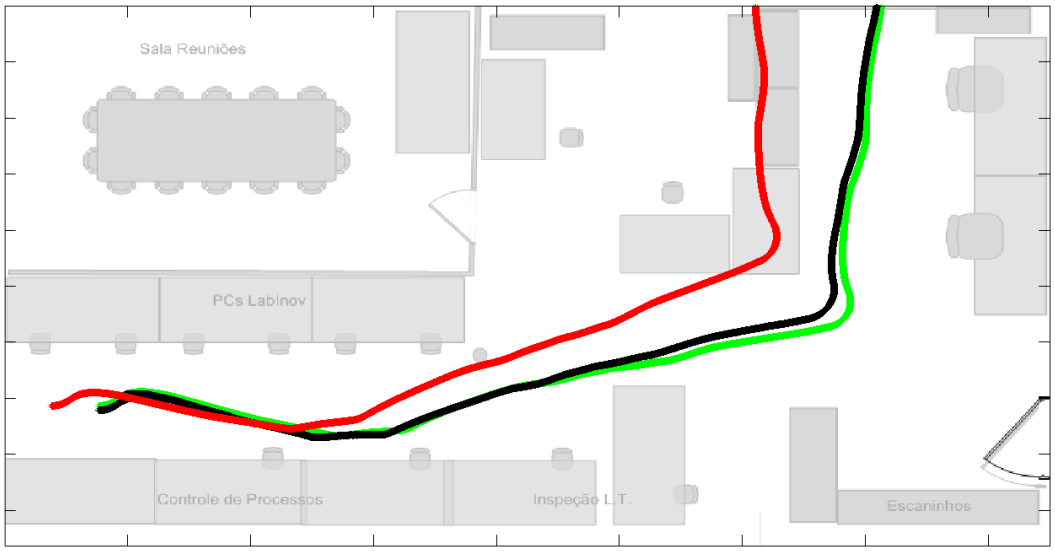


Figura 4.21: Resultado utilizando odometria sem correção (em vermelho), resultado utilizando *ARToolKit* (preto) e resultados utilizando o filtro com RFID (verde).

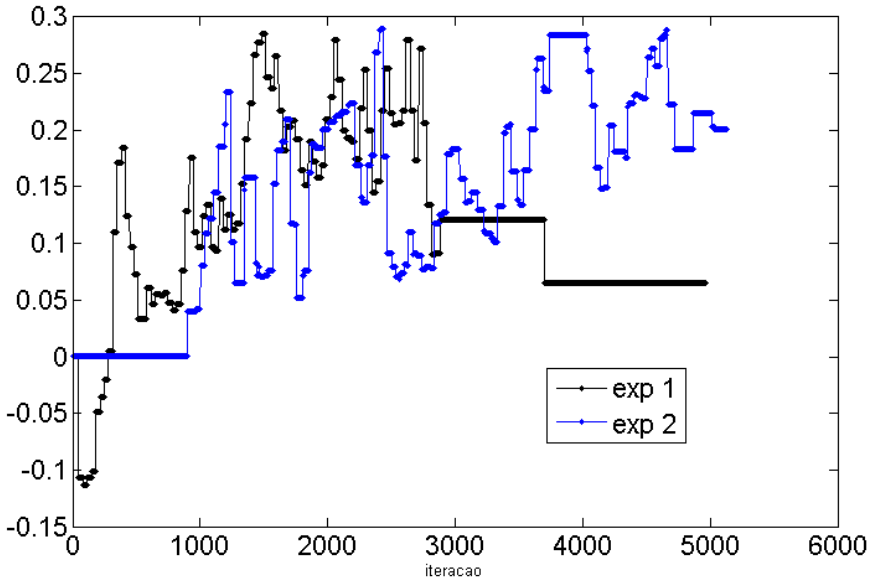


Figura 4.22: Variação de α_0 durante o percurso com controle de trajetória para 2 experimentos diferentes.

Capítulo 5

Conclusões

A proposta principal para o trabalho era uma contribuição com o projeto de localização utilizando RFID desenvolvido no LARA atualmente. A contribuição foi feita em diversos aspectos. A principal foi um sistema de localização embarcado no *Aramis*, o robô móvel. Sistema esse que tem uma alta precisão ao se utilizar filtragem estocástica e um sensor de câmera. A localização está funcinando em sistema operacional Linux tempo real, sincronizado com a aquisição de dados RFID, permitindo testes com a temporização garantida. Ao fazer uso do Filtro de Kalman *Unscented*, foi possível fazer a correção da posição do robô, mesmo utilizando um sistema com os problemas encontrados no *ARToolKit*, como qualidade de câmeras, e problemas ligados ao próprio sistema e condições ambientes desfavoráveis.

Foi desenvolvido um controlador de trajetórias, permitindo que, ao se definir pontos no laboratório, o robô repetisse o experimento diversas vezes. O controle de trajetória também está embarcado no *Aramis*, viabilizando que futuros experimentos façam um mesmo percurso repetidas vezes, em condições diferentes, para este ou outros trabalhos. Quanto ao controle de trajetória, a plataforma ainda tiraria proveito de uma análise mais profunda quanto a estabilidade com respeito aos ganhos do controlador proporcional como uma análise pelo método de Lyapunov. Outro trabalho interessante seria o levantamento de planos de fase para caracterizar graficamente erros e atratores. Ambas as análises não foram executadas no presente trabalho, mas são encorajadas para trabalhos futuros.

Além da contribuição na plataforma de aquisição de dados, *Aramis*, foi realizado um experimento em ambiente externo (no campo de futebol do Centro Olímpico da Universidade de Brasília), para a caracterização do equipamento utilizado. Dessa forma, houve um melhor entendimento do funcionamento do conjunto disponível. Assim, considera-se conhecida a caracterização do modelo de RFID por distâncias para ambientes que se aproximam do ideal.

Foi feita proposta para a caracterização do sinal em ambientes internos gerais. Apesar do modelo ainda não ter sido o ideal, foi visto que, com melhorias de modelo, é possível utilizar o RFID para a correção da posição do robô móvel. Há uma dificuldade em se modelar a atenuação do sinal provocada por razões imprevisíveis no laboratório. Embora esse modelamento poça ser buscado em um trabalho focado em propagação de ondas, é feita aqui a sugestão de um modelo

que guarde informação a priori da configuração de atenuação do ambiente de trabalho específico. Como foi considerado no desenvolvimento, diversas técnicas poderiam ser implementadas para se alcançar essa condição. Como sugestão para trabalhos futuros, poderia ser feita uma "calibração" do ambiente tomando valores de RSSI em posições conhecidas e levantando um mapa de atenuação do ambiente. Averiguada a necessidade, seria possível incrementar o modelo com um mapa de atenuações associado a um algoritmo de atualização, conferindo maior plásticidade ao sistema. Por último, uma solução que poderia ser implementada a parte ou combinada com as já citadas seria a de utilização de *tags* fixas em posições conhecidas. Embora se saiba que o equipamento disponível é de natureza ativa, e o sinal varie conforme a bateria de cada *tag*, esse problema poderia ser contornado (talvez ao preço de se caracterizar o sinal de cada *tag* fixa utilizada) oferecendo um mapa de atenuação sempre calibrado.

Além de uma melhora na modelagem do sinal RFID, uma das propostas é utilizar informações do ambiente para poder melhorar a localização. Tendo informação prévia do ambiente, como é o caso por exemplo do mapa do LARA, podem ser feitas restrições de posição e trajetória que minimizam chances de erro, pelo menos no que diz respeito à localização de uma *tag* quanto a situar-se ou não em ambientes (no caso do LARA, possíveis ambientes seriam sala de reuniões, sala de *hardware*, sala comum, etc.). O uso da odometria na utilização de RFID com robô móvel insere uma restrição dada pelo modelo cinemático do robô associado aos dados dos *encoders*. O modelo de evolução do robô foi muito importante para a boa estimação das trajetórias percorridas pelo durante os testes. Embora seja menos determinístico, um modelo de evolução da posição de indivíduos deveria ser formulado para a utilização como um sistema para localização de pessoas. Na linha de robótica móvel, mais informação de sensores (como os sonares da plataforma *Aramis*) trariam ainda maior confiabilidade à localização.

Um ponto que merece destaque foi a investigação do uso de um sistema de realidade aumentada para localização tridimensional. Há linhas de pesquisa em fase inicial de desenvolvimento no LARA com utilização da mesma biblioteca como ferramente (*ARToolKit*) para auxiliar na guiagem de veículos aéreos. Uma grande dificuldade de se obter resultados acurados com os algoritmos de realidade aumentada implementados está no fato de que, em geral, ao se fixar uma marca numa posição conhecida, eventuais pequenos erros na estimação da orientação relativa da marca em relação ao usuário podem se refletir em erros significativos de posição medidos a depender da distância em que se está do alvo. Para anular ou minimizar esses efeitos, seria útil que se pudesse inferir alguma informação entre alvo e ambiente, além de simplesmente assumir a restrição de montagem do alvo no ambiente. Assim possiveis más identificações de orientação relativa poderiam ser corrigidas. Um projeto com essas características é desenvolvido pelo Laboratório de Tecnologia de Interface Humana na Universidade de Washington. Nele, diversas marcas são presas a um quadro branco e a restrição física de alinhamento das marcas é utilizada para obter um estimativa otimizada da posição e orientação do quadro branco com relação ao usuário. Como analisado no capítulo de desenvolvimento, é necessário que se leve em conta um compromisso entre número de marcas utilizadas e tamanho das marcas. Contudo, é encorajado o uso de multimarcas para se inserir parametros extrínsecos entre marcas para obtenção de melhores resultados tridimensionais.

Em linhas gerais foram desenvolvidas satisfatóriamente as contribuições propostas ao projeto

em questão. Além do substrato e ferramentas desenvolvidas para realização de testes, foram iniciados alguns ensaios com RFID e uma linha de pensamento para a condução do projeto é sugerida. Os resultados com os testes de RFID, carecem de uma maior análise em termos de subestimação ou superestimação das matrizes de covariância associadas a cada fase da filtragem. Na verdade esses ensaios tiveram o propósito de levantar dados e hipóteses que seriam parte do início de um novo desenvolvimento. Os resultados obtidos são considerados animadores, e figuram no manuscrito como uma forma de motivação para a continuidade do desenvolvimento.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] WAN, E. A.; MERWE, R. V. D.; NELSON, A. T. Dual Estimation and the Unscented Transformation. In: *in Neural Information Processing Systems*. Denver, CO, USA: MIT Press, 2000. p. 666–672.

[2] MALBEZIN, P.; PIEKARSKI, W.; THOMAS, B. H. Measuring ARToolKit Accuracy in Long Distance Tracking Experiments. 2002.

[3] BAJAJ, R.; RANAWEERA, S.; AGRAWAL, D. GPS: Location-Tracking Technology. *Computer*, v. 35, n. 4, p. 92 –94, apr 2002. ISSN 0018-9162.

[4] HUANG, J. yao; TSAI, C.-H. Improve GPS positioning accuracy with context awareness. In: *Ubi-Media Computing, 2008 First IEEE International Conference on*. Lanzhou, China: IEEE, 2008. p. 94 –99.

[5] DIGGELEN, F. van. Indoor GPS theory implementation. In: *Position Location and Navigation Symposium, 2002 IEEE*. Palm Springs, California,: IEEE Press, 2002. p. 240 – 247.

[6] NI, L. et al. LANDMARC: indoor location sensing using active RFID. In: *Pervasive Computing and Communications, 2003. (PerCom 2003). Proceedings of the First IEEE International Conference on*. Fort Worth, Texas: IEEE Press, 2003. p. 407 – 415.

[7] ZHAO, Y.; LIU, Y.; NI, L. VIRE: Active RFID-based Localization Using Virtual Reference Elimination. In: *Parallel Processing, 2007. ICPP 2007. International Conference on*. XiAn, China: IEEE Press, 2007. p. 56. ISSN 0190-3918.

[8] JIANG, X.; LIU, Y.; WANG, X. An Enhanced Approach of Indoor Location Sensing Using Active RFID. In: *Information Engineering, 2009. ICIE '09. WASE International Conference on*. Taiyuan, Shanxi, China: IEEE Press, 2009. v. 1, p. 169 –172.

[9] HUANG, X.; JANASWAMY, R.; GANZ, A. Scout: Outdoor Localization Using Active RFID Technology. In: *Broadband Communications, Networks and Systems, 2006. BROADNETS 2006. 3rd International Conference on*. San José, CA, USA: IEEE, 2006. p. 1 –10.

[10] BEKKALI, A.; SANSON, H.; MATSUMOTO, M. RFID Indoor Positioning Based on Probabilistic RFID Map and Kalman Filtering. In: *Wireless and Mobile Computing, Networking and Communications, 2007. WiMOB 2007. Third IEEE International Conference on*. White Plains, NY: IEEE, 2007. p. 21.

[11] FIGUEREDO, L.; COUTO, F.; BAUCHSPIESS, A. An Evaluation of RSSI Based Indoor Localization Systems in Wireless Sensor Networks. In: *IIX Simpósio Brasileiro de Automação Inteligente (SBAI 2009)*. Brasília: [s.n.], 2009.

[12] FONSECA, L. O. da. *Sistema de Localização RFID de Usuários Visando a Racionalização de Energia em Ambientes Inteligentes*. Brasília, fev. 2011. Trabalho de Graduação.

[13] MCKERROW, P. J. *Introduction to Robotics*. Wokingham: Addison Wesley, 1990. ISBN 0201182408.

[14] JAZWINSKI, A. H. *Stochastic Processes and Filtering Theory*. New York: Academic Press, 1970. ISBN 9788570415844.

[15] AGUIRRE, L. A. *Introdução à Identificação de Sistemas: Técnicas lineares e não lineares aplicadas à sistemas reais*. Belo Horizonte: Editora UFMG, 2007. ISBN 9788570415844.

[16] JULIER, S. J.; UHLMANN, J. K. A New Extension of the Kalman Filter to Nonlinear Systems. In: . [S.l.: s.n.], 1997. p. 182–193.

[17] JULIER, S.; UHLMANN, J. K. *A General Method for Approximating Nonlinear Transformations of Probability Distributions*. [S.l.], 1996.

[18] Santana, P. H. R. Q. A. *Filtragem Estocástica para Sistemas Híbridos e suas Aplicações em Robótica Aérea*. Dissertação (Mestrado) — Universidade de Brasília, Feb. 2011.

[19] JULIER, S. J.; UHLMANN, J. K.; HUGH. A New Approach for Filtering Nonlinear Systems. In: *Proceedings of the ACC'95*. [S.l.: s.n.], 1995. v. 3, p. 1628–1632.

[20] THRUN, S.; BURGARD, W.; FOX, D. *Probabilistic Robotics (Intelligent robotics and autonomous agent series)*. Cambridge: The MIT Press, 2005. Hardcover. ISBN 9780262201629.

[21] BORGES, G. A.; LIMA, A. M. N.; DEEP, G. S. *Design Of An Output Feedback Trajectory Controller For An Automated Guided Vehicle*. 2000.

[22] MOBILEROBOTS INC. *Pioneer 3 Operations Manual*. [S.l.], 2006.

[23] PIEKARSKY, W. *Interactive 3D Modelling in Outdoor Augmented Reality Worlds*. Tese (Doutorado) — University of South Australia, Adelaide, Australia, 2004.

[24] KATO, H.; BILLINGHURST, M. Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System. *Augmented Reality, International Workshop on*, IEEE Computer Society, Los Alamitos, CA, USA, v. 0, p. 85, 1999.

[25] AND, W. P. *Using ARToolKit for 3D Hand Position Tracking in Mobile Outdoor Environments*. 2002.

[26] ABDULLAH, J.; MARTINEZ, K. Camera Self-Calibration for the ARToolKit.

[27] NATH, B.; REYNOLDS, F.; WANT, R. RFID Technology and Applications. *Pervasive Computing, IEEE*, v. 5, n. 1, p. 22 – 24, jan.-march 2006. ISSN 1536-1268.

[28] SHENG, Q. Z.; LI, X.; ZEADALLY, S. Enabling Next-Generation RFID Applications: Solutions and Challenges. *Computer*, v. 41, n. 9, p. 21 –28, sept. 2008. ISSN 0018-9162.

ANEXOS

I. ESTIMAÇÃO DE PARÂMETROS

I.1 Mínimos Quadrados

O objetivo dessa seção é apresentar o Método de Mínimos Quadrados (MMQ), muito utilizado na identificação de parâmetros. Em [15], o autor mostra a teoria do MMQ.

Considerando uma função da forma $y = f(x)$, essa função é caracterizada por um vetor com n_θ parâmetros θ. O objetivo do MMQ é obter o vetor de parâmetros θ a partir de medidas de x e de y, ou seja $\{x_1, x_2, \cdots, x_N\}$ e $\{y_1, y_2, \cdots, y_N\}$.

Considerando a mesma aplicação em N pontos, como mostrado na Eq. I.1, é esperado que tal função dependa dos parâmetros, e possa ser escrita parametrizada por θ, como $y = (x, \theta)$. Sendo conhecidas as N observações de y e de x, ou seja, $\{y_1, y_2, \cdots, y_N\}$ e $\{x_1, x_2, \cdots, x_N\}$, é possível obter os parâmetros θ que represente melhor essa função, considerando que essa função seja invariante no tempo.

$$\begin{aligned} y_1 &= f(x_1) \\ y_2 &= f(x_2) \\ &\vdots \\ y_N &= f(x_N) \end{aligned} \tag{I.1}$$

A função $y = f(x)$ pode ser reescrita como mostrado na Eq. I.2, onde o vetor X é chamado de vetor de regressores, e o vetor Y é o vetor das N observações de saída.

$$Y = X\theta \tag{I.2}$$

Caso a matriz X não seja singular, e seja quadrada, e $N = n_\theta$, é possível determinar o vetor de parâmetros θ como mostrado na Eq. I.3

$$\theta = X^{-1}Y \tag{I.3}$$

Se o número de medidas N for maior do que o número de parâmetros, o sistema é sobredeterminado. O objetivo é que seja encontrado θ que seja ótima para o sistema, minimizando o erro quadrático médio. Considerando que o sistema tenha os parâmetros $\theta = \hat{\theta}$, sendo $\hat{\theta}$ já conhecido. A Eq. I.4 mostra a configuração de tal sistema, sendo que o termo ξ representa o erro de tal modelagem.

$$y = X^T\hat{\theta} + \xi \tag{I.4}$$

É interessante que $\hat{\theta}$ seja tal que minimize ξ. Um bom índice que quantifica a estimação dos parâmetros é o erro quadrático, assim, $\hat{\theta}$ deve ser tal que minimize a Eq. I.5.

$$\begin{aligned} J_{MQ} &= \left(y - X\hat{\theta}\right)^T \left(y - X\hat{\theta}\right) \\ &= y^T y - y^T X\hat{\theta} - \hat{\theta}^T X^T y + \hat{\theta}^T X^T X\hat{\theta} \end{aligned} \tag{I.5}$$

Derivando a Eq. I.5 em relação a $\hat{\theta}$ é encontrada a Eq. I.6.

$$\begin{aligned} \frac{\partial J_{MQ}}{\partial \hat{\theta}} &= -\left(y^T X\right)^T - X^T y + \left(X^T X + X^T X\right)\hat{\theta} \\ &= -X^T y - X^T y + 2X^T X\hat{\theta} \end{aligned} \tag{I.6}$$

Igualando a Eq. I.6 a 0, é encontrado $\hat{\theta}$ tal que minimize o erro, como mostrado na Eq. I.7.

$$\hat{\theta} = \left[X^T X\right]^{-1} X^T y \tag{I.7}$$

Para que o $\hat{\theta}$ encontrado seja um ponto de mínimo da função, é necessário que a derivada segunda de J_{MQ} seja maior que zero, como mostrado na Eq. I.8.

$$\frac{\partial^2 J_{MQ}}{\partial \theta^2} = 2XX^T > 0 \tag{I.8}$$

O MMQ como foi apresentado permite apenas a identificação por batelada, ou seja, os dados são coletados, e depois é realizado o algoritmo. Para identificar os parâmetros *online* é necessário que seja utilizada outra técnica, como por exemplo Mínimos Quadrados Recursivos.

I.2 Taxa de Redução de Erro

Para a utilização do MMQ, é suposto que já se conheça sobre a estrutura do modelo, ou seja, os regressores utilizados e suas combinações. O critério da Taxa de Redução de Erro (ERR) permite que dentre uma possibilidade de regressores, sejam determinados aqueles que têm maior influência no modelo. Assim, com essa técnica, é possível que os parâmetros sejam determinados.

É considerado que o modelo do sistema seja escrito como motrado na Eq. I.9, sendo $y(k)$ a *k-ésima* observação, $\psi(k-1)$ os regressores do sistema, $\hat{\theta}$ o vetor dos parâmetros e $\xi(k)$ o erro.

$$\begin{aligned} y(k) &= \psi(k)\hat{\theta} + \xi(k) \\ &= \sum_{i=1}^{n_\theta} \hat{\theta}_i \psi_i(k-1) + \xi(k) \end{aligned} \tag{I.9}$$

Ortogonalizando o vetor de regressores, é possível reescrever a Eq. I.9 como mostrado na Eq. I.10, sendo $\hat{g}_i$ definido na Eq. I.11.

$$y(k) = \sum_{i=1}^{n_\theta} \hat{g}_i w_i(k-1) + \xi(k) \tag{I.10}$$

$$\hat{g}_i = \frac{\langle w_i, y \rangle}{\langle w_i, w_i \rangle},\ i = 1, \cdots, n_\theta \tag{I.11}$$

Dessa forma, os regressores w_i são ortogonais, ou seja, a média temporal $w_i(k)\, w_j(k)$ é nula. Dessa forma, a soma dos valores quadráticos de $y(k)$ é mostrada na Eq. I.12.

$$\begin{array}{rcl} \sum_{k=1}^{N} y(k)^2 & = & \langle y, y \rangle \\ & = & \sum_{i=1}^{n_\theta} \hat{g}_i^2 \langle w_i, w_i \rangle + \langle \xi, \xi \rangle \end{array} \tag{I.12}$$

Assim, a soma de valores quadráticos de $y(k)$ pode ser explicada como a soma de valores quadráticos de cada regressor. A parcela não explicada é atribuida a ξ, soma dos valores quadráticos dos resíduos. Dessa forma, é possível quantificar a contribuição de cada uma dos regressores considerados para o valor final de $\langle y, y \rangle$. A taxa de redução de erro de cada um dos i regressores é mostrada na Eq. I.13.

$$\mathrm{ERR}_i = \frac{\hat{g}_i^2 \langle w_i, w_i \rangle}{\langle y, y \rangle} \tag{I.13}$$

O ERR permite indicar a contribuição de cada regressor para a explicação da obervação. Essa técnica pode ser utilizada em modelos (N)ARX e (N)ARMAX. Para ortogonalizar os vetores existem diversos algoritmos, recomenda-se a referência [15] para maiores informações.

II. ALVOS UTILIZADOS

A seguir são apresentados os alvos utilizados para localização do robô móvel.

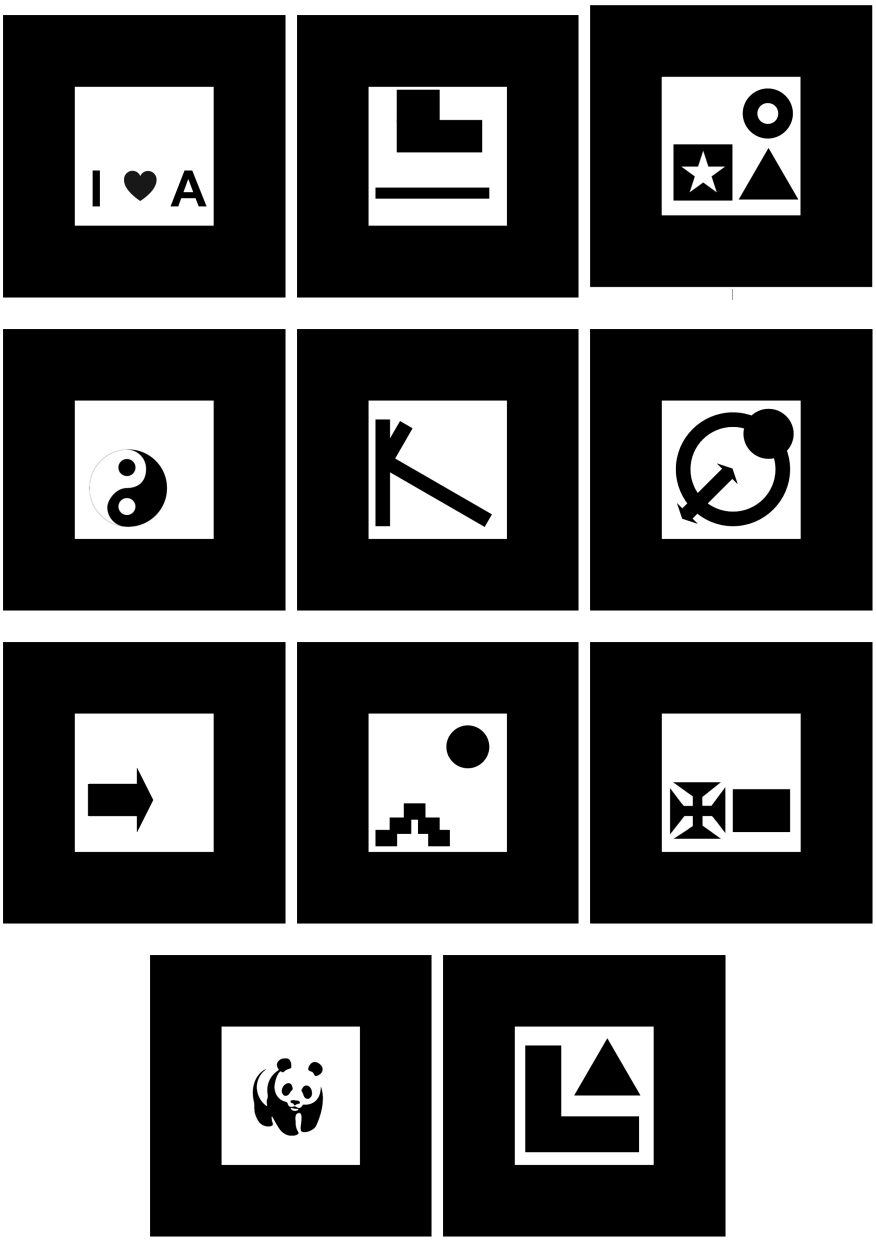


Figura II.1: Alvos utilizados

III. DESCRIÇÃO DO CONTEÚDO DO CD

O CD possui 3 pastas. A pasta *Relatorio* apresenta uma cópia do manuscrito em formato digital. A pasta *Resumo* apresenta um resumo e o conjunto de palavras chave do trabalho, para consultas breves. A pasta *Arquivos* conta com todo o código produzido (incluindo *scripts* em MATLAB e códigos em linguagem *C* e *C++*), arquivos de imagens para alvos utilizados, vídeos demonstrativos produzidos e conjuntos de dados utilizados para análise, tudo organizado em subpastas.