

CENTRO UNIVERSITÁRIO FEI

JOÃO VICTOR LOURENÇO AGUIAR

**FUSÃO DE SENsoRES PARA ESTIMATIVA DE LOCALIZAÇÃO DE ROBÔS NO  
AMBIENTE DA ROBOCUP SMALL SIZE LEAGUE**

São Bernardo do Campo

2025

JOÃO VICTOR LOURENÇO AGUIAR

**FUSÃO DE SENsoRES PARA ESTIMATIVA DE LOCALIZAÇÃO DE ROBÔS NO  
AMBIENTE DA ROBOCUP SMALL SIZE LEAGUE**

Dissertação de Mestrado apresentada ao Centro Universitário da FEI para obtenção do título de Mestre em Engenharia Elétrica, orientado pelo Prof. Dr. Flavio Tonidandel.

São Bernardo do Campo

2025

## **AGRADECIMENTOS**

Eu gostaria de agradecer à minha família, especialmente pais e irmão, por todo apoio durante a jornada e pelo suporte para que eu pudesse focar nos estudos.

Aos meus amigos, que também me apoiaram e motivaram durante essa etapa.

Ao meu orientador, Prof. Dr. Flavio Tonidandel, que me ajudou desde o começo da graduação quando ingressei no projeto RoboFEI e me ajudou a solucionar dúvidas, realizar discussões e providenciando orientações ao longo do caminho.

A todos os membros que convivi durante esse tempo na equipe SSL do RoboFEI, além dos professores Plinio Thomaz Aquino Junior e Reinaldo Bianchi. Observei muito conhecimento convivendo com todos no dia-a-dia e durante as competições que participei.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

“Alguns homens vêem as coisas como são, e dizem ‘Por quê?’. Eu sonho com as coisas que nunca foram e digo ‘Por que não?’”

George Bernard Shaw

## RESUMO

Este trabalho propõe o estudo de sensores para localização de robôs móveis no ambiente de futebol de robôs da categoria *Small Size League* (SSL) da RoboCup. Para isso, foram comparadas diferentes combinações de sensores, sendo eles: encoders acoplados aos eixos das rodas, sensor *Inertial Measurement Unit* (IMU) composto por giroscópio e acelerômetro, e câmeras externas pertencentes ao sistema compartilhado de visão da liga (SSL-Vision). O sistema é baseado em fusão de sensores, utilizando o EKF (Extended Kalman Filter). A seleção dos sensores baseou-se em uma análise dos trabalhos recentes desenvolvidos na categoria SSL, identificando os sensores mais utilizados pelas equipes de ponta. Os sensores escolhidos foram testados utilizando um robô real da equipe RoboFEI, em dois cenários diferentes no campo de testes da equipe. O primeiro, localizado na região central do campo, sofre com sobreposição de câmeras e consequente duplicação de leituras. O segundo, de maiores dimensões e localizado nas bordas do campo, é afetado por distorções causadas pelas lentes olho de peixe. Para a avaliação, foram utilizadas duas métricas: o erro médio de distância em relação a um sistema de ground truth e o tempo de atualização das estimativas de posição. Os sensores foram combinados dois a dois entre as fases de predição e correção do EKF. O sistema de ground truth foi desenvolvido especificamente para este trabalho, utilizando dois LiDARs externos ao robô e a transformada de Hough para identificação do padrão circular de um típico robô da categoria SSL em uma imagem gerada com os pontos dos LiDARs. No primeiro teste, foi possível observar que as melhores combinações, em termos de erro médio de distância, utilizavam os dados do SSL-Vision na fase de correção, enquanto na fase de predição usavam leituras de sensores em vez do modelo cinemático do robô omnidirecional, evidenciando que os dados de sensores ajudam a estimar melhor a posição do robô. Quanto ao tempo de atualização, como os cálculos do EKF para fusão de sensores são realizados de forma embarcada no microcontrolador do robô, notou-se que a fusão de sensores é cinco vezes mais rápida do que o sistema atualmente utilizado pela equipe. No segundo teste, foi notável que, nas combinações sem o SSL-Vision, tanto o erro médio de distância quanto os valores máximos aumentaram em relação ao primeiro teste. Isso evidencia um aumento do erro em trajetos maiores, além da necessidade de utilização de um sensor que forneça observações absolutas do robô dentro do ambiente. Por fim, com este trabalho foi possível realizar a comparação da utilização de diferentes sensores no ambiente de futebol de robôs, em um sistema de fusão de sensores para localização utilizando o EKF de forma embarcada. Conclui-se que, para a liga SSL, a utilização dos dados do SSL-Vision é inevitável

para a correção da posição do robô, devido à precisão necessária na realização das jogadas durante a partida. No entanto, ao fundir sensores como IMU e encoders, foi possível obter estimativas de posição ligeiramente melhores, com tempos de atualização significativamente menores.

Keywords: Localização. Predição. Correção. Sensores. Embarcado.

## ABSTRACT

This work proposes the study of sensors for mobile robot localization in the robot soccer environment of the *Small Size League* (SSL) category of RoboCup. To this end, different combinations of sensors were compared, namely: encoders attached to the wheel shafts, an *Inertial Measurement Unit* (IMU) sensor composed of a gyroscope and an accelerometer, and external cameras that are part of the league’s shared vision system (SSL-Vision). The system is based on sensor fusion using the Extended Kalman Filter (EKF). The selection of sensors was based on an analysis of recent works developed in the SSL category, identifying the most commonly used sensors by top-performing teams. The selected sensors were tested using a real robot from the RoboFEI team in two different scenarios on the team’s test field. The first scenario, located in the central region of the field, is affected by overlapping cameras, which causes position divergence due to duplicated readings. The second scenario, with larger dimensions and located near the edges of the field, is influenced by distortions caused by fisheye lenses. Two metrics were used for the evaluation: the average distance error relative to a ground truth system and the update time of position estimates. The sensors were combined in pairs between the prediction and correction phases of the EKF. The ground truth system was specifically developed for this work, using two LiDARs external to the robot and the Hough transform to identify the circular pattern of a typical SSL robot in an image generated from the LiDAR point cloud. In the first test, it was observed that the best combinations in terms of average distance error used SSL-Vision data during the correction phase, while using sensor readings—rather than the kinematic model of the omnidirectional robot—in the prediction phase, highlighting that sensor data helps provide better position estimates. Regarding update time, since the EKF calculations for sensor fusion are executed onboard the robot’s microcontroller, it was found that the fusion process is five times faster than the system currently used by the team. In the second test, it was notable that combinations without SSL-Vision showed increases in both average distance error and maximum error values compared to the first test. This indicates a rise in error over longer trajectories, as well as the need for a sensor capable of providing absolute observations of the robot within the environment. Finally, this work enabled a comparison of different sensors within the robot soccer environment, in a sensor fusion system for localization using an embedded EKF. It was concluded that, for the SSL league, the use of SSL-Vision data is essential for correcting the robot’s position due to the level of precision required for executing plays during a match. However, by fusing

sensors such as IMU and encoders, it was possible to achieve slightly better position estimates with significantly lower update times.

Keywords: Localization. Prediction. Correction. Sensors. Embedded.

## LISTA DE ILUSTRAÇÕES

Figura 1 – Ilustração de uma partida da SSL. . . . .	21
Figura 2 – Imagem de um robô SSL da equipe RoboFEI. . . . .	22
Figura 3 – Comparação da velocidade angular adquirida pelo sistema de visão global e pelo giroscópio. . . . .	24
Figura 4 – Padrões de cores para identificação dos robôs. . . . .	28
Figura 5 – Ilustração da montagem e funcionamento de um encoder óptico. . . . .	29
Figura 6 – Diagrama representando a diferença entre as fusões complementar, competitiva e cooperativa. . . . .	45
Figura 7 – Diagrama representando a diferença entre as fusões baseadas no nível de abstração dos dados. . . . .	47
Figura 8 – Diagrama relacionando as classificações três níveis e Dasarathy. . . . .	48
Figura 9 – Exemplo de distribuição gaussiana. . . . .	51
Figura 10 – Ciclo do filtro de Kalman discreto. . . . .	53
Figura 11 – Exemplo de predição e atualização das covariâncias do filtro de Kalman. .	56
Figura 12 – Exemplo de predição e atualização das covariâncias do filtro de Kalman. .	57
Figura 13 – Visão geral da fusão de sensores da odometria com o sistema de visão da categoria MSL. . . . .	62
Figura 14 – Placa de desenvolvimento STM32F411E-Disco utilizada. . . . .	69
Figura 15 – Diagrama de implementação da fusão de sensores para localização. . . . .	72
Figura 16 – Representação do posicionamento dos lasers em campo. . . . .	74
Figura 17 – 10000 amostras dos dados do acelerômetro antes da calibração. . . . .	75
Figura 18 – Esfera de calibração do acelerômetro. . . . .	76
Figura 19 – 10000 amostras dos dados do acelerômetro depois da calibração. . . . .	77
Figura 20 – 10000 amostras dos dados do giroscópio antes da calibração. . . . .	79
Figura 21 – 10000 amostras dos dados do giroscópio depois da calibração. . . . .	80
Figura 22 – Ilustração do teste de quadrado menor. . . . .	81
Figura 23 – Problema de sobreposição de câmeras. . . . .	82
Figura 24 – Ilustração do teste de quadrado maior. . . . .	83
Figura 25 – Problema da distorção da imagem de câmeras com lente olho de peixe. .	83

Figura 26 – Arquitetura do software. . . . .	86
Figura 27 – Duas etapas da arquitetura desenvolvida. . . . .	88
Figura 28 – Cenários de testes do sistema de ground truth. . . . .	89
Figura 29 – Diferença entre os eixos do campo e do robô. . . . .	94
Figura 30 – Diferença das velocidades calculadas pelo acelerômetro e pelos encoders. .	99
Figura 31 – Acelerações obtidas durante o teste. . . . .	99
Figura 32 – Erro médio de distância no teste com trajeto do quadrado menor. . . . .	104
Figura 33 – Erro máximo no teste com trajeto do quadrado menor. . . . .	105
Figura 34 – Erro médio de orientação no teste com trajeto do quadrado menor. . . . .	106
Figura 35 – Erro médio de distância no teste com trajeto do quadrado maior. . . . .	108
Figura 36 – Erro máximo no teste com trajeto do quadrado maior. . . . .	109
Figura 37 – Erro médio de orientação no teste com trajeto do quadrado maior. . . . .	110
Figura 38 – Orientação ao longo de uma amostra do Cenário 5. . . . .	111
Figura 39 – Orientação ao longo de uma amostra do Cenário 7. . . . .	111
Figura 40 – Teste de 8 segundos sem visão para o Cenário 4. . . . .	112
Figura 41 – Teste de 8 segundos sem visão para o Cenário 6. . . . .	113

## LISTA DE TABELAS

Tabela 1 – Combinações dos sensores na predição e correção do Filtro de Kalman. . . . .	70
Tabela 2 – Especificações de hardware do notebook. . . . .	71
Tabela 3 – Especificações dos LiDARs utilizados. . . . .	73
Tabela 4 – Análise estatística dos dados do acelerômetro antes da calibração. . . . .	76
Tabela 5 – Análise estatística dos dados do acelerômetro depois da calibração. . . . .	77
Tabela 6 – Análise estatística dos dados do giroscópio antes da calibração. . . . .	79
Tabela 7 – Análise estatística dos dados do giroscópio depois da calibração. . . . .	81
Tabela 8 – Resultados alcançados para o eixo X no campo de teste. . . . .	90
Tabela 9 – Resultados alcançados para o eixo Y no campo de teste. . . . .	90
Tabela 10 – Erro e desvio padrão médios nos cenários. . . . .	91
Tabela 11 – Resultados dos quadrantes para o eixo X. . . . .	91
Tabela 12 – Resultados dos quadrantes para o eixo Y. . . . .	92
Tabela 13 – Resumo dos resultados obtidos dos cenários no teste do quadrado menor. .	107
Tabela 14 – Resumo dos resultados obtidos dos cenários no teste com quadrado maior.	110

## **LISTA DE ALGORITMOS**

Algoritmo 1 – Filtro de Kalman linear . . . . .	55
Algoritmo 2 – Filtro de Kalman estendido . . . . .	58

## LISTA DE ABREVIATURAS

DC	<i>Direct Current</i>
EKF	Filtro de Kalman Estendido
FP	Filtro de Partículas
FSEV	Filtro Suave de Estrutura Variável
GPS	<i>Global Positioning System</i>
HT	Transformada de Hough
I <sup>2</sup> C	<i>Inter-Integrated Circuit</i>
IMU	<i>Inertial Measurement Unit</i>
INS	<i>Inertial Navigation System</i>
KF	Filtro de Kalman
KG	Ganho de Kalman
LiDAR	<i>Light Detection And Ranging</i>
MEMS	Sistema Microeletromecânico
MSL	<i>Middle Size League</i>
RMSE	Erro Quadrático Médio
SPI	<i>Serial Peripheral Interface</i>
SSL	<i>Small Size League</i>
TDP	<i>Team Description Paper</i>
UKF	Filtro de Kalman <i>Unscented</i>

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	16
1.1	OBJETIVO	17
1.2	JUSTIFICATIVA	18
1.3	ESTRUTURA DA DISSERTAÇÃO	18
1.4	CONTRIBUIÇÕES	19
<b>1.4.1</b>	<b>Publicações</b>	19
<b>1.4.2</b>	<b>Materiais eletrônicos</b>	19
<b>2</b>	<b>FUTEBOL DE ROBÔS</b>	20
2.1	<i>SMALL SIZE LEAGUE</i>	20
<b>2.1.1</b>	<b>Sistema de visão da categoria SSL</b>	22
<b>2.1.2</b>	<b>Fusão de sensores na <i>Small Size League</i></b>	24
2.2	SENSORES	25
<b>2.2.1</b>	<b>Tipos de sensores</b>	26
<b>2.2.2</b>	<b>Sistema de câmeras</b>	27
<b>2.2.3</b>	<b>Encoder</b>	28
<b>2.2.4</b>	<b>Giroscópio</b>	30
<b>2.2.5</b>	<b>Acelerômetro</b>	30
<b>2.2.6</b>	<b><i>Inertial Measurement Unit (IMU)</i></b>	32
<b>2.2.7</b>	<b>Calibração dos sensores</b>	32
<b>2.2.7.1</b>	<b><i>Calibração do acelerômetro</i></b>	33
<b>2.2.7.2</b>	<b><i>Calibração do giroscópio</i></b>	35
<b>3</b>	<b>LOCALIZAÇÃO DE ROBÔS MÓVEIS</b>	36
3.1	O PROBLEMA DA LOCALIZAÇÃO	36
3.2	OS DESAFIOS DA LOCALIZAÇÃO	37
3.3	INFORMAÇÃO DISPONÍVEL	38
3.4	FUSÃO DE SENSORES	40
<b>3.4.1</b>	<b>Vantagens e Desvantagens</b>	41
<b>3.4.2</b>	<b>Classificação de técnicas</b>	43
<b>3.4.2.1</b>	<b><i>Classificação baseada na relação entre as fontes de dados</i></b>	43
<b>3.4.2.2</b>	<b><i>Classificação em três níveis</i></b>	44
<b>3.4.2.3</b>	<b><i>Classificação baseada na entrada e saída do sistema</i></b>	45

3.5	FILTRO DE KALMAN . . . . .	46
3.5.1	<b>Utilização</b> . . . . .	47
3.5.2	<b>Premissas</b> . . . . .	49
3.5.2.1	<i>Sistema dinâmico linear</i> . . . . .	49
3.5.2.2	<i>Características do ruído</i> . . . . .	50
3.5.2.3	<i>Processo a ser estimado</i> . . . . .	51
3.5.3	<b>Equações</b> . . . . .	52
3.5.3.1	<i>Predição</i> . . . . .	52
3.5.3.2	<i>Correção</i> . . . . .	54
3.6	FILTRO DE KALMAN ESTENDIDO . . . . .	55
4	<b>TRABALHOS RELACIONADOS</b> . . . . .	59
5	<b>METODOLOGIA</b> . . . . .	68
5.1	DOMÍNIO DE TESTES . . . . .	68
5.2	COMBINAÇÃO DOS SENSORES . . . . .	69
5.3	IMPLEMENTAÇÃO . . . . .	70
5.4	<i>GROUND TRUTH</i> . . . . .	72
5.5	SENSORES INERCIAIS . . . . .	73
5.5.1	<b>Acelerômetro</b> . . . . .	74
5.5.1.1	<i>Calibração</i> . . . . .	75
5.5.2	<b>Giroscópio</b> . . . . .	78
5.5.2.1	<i>Calibração</i> . . . . .	78
5.6	TESTES . . . . .	81
5.7	AVALIAÇÃO . . . . .	82
6	<b>RESULTADOS</b> . . . . .	85
6.1	AQUISIÇÃO DO GROUND TRUTH . . . . .	85
6.1.1	<b>Técnica utilizada e visão geral do sistema</b> . . . . .	85
6.1.2	<b>Resultados do sistema desenvolvido</b> . . . . .	88
6.2	IMPLEMENTAÇÃO DOS CENÁRIOS . . . . .	92
6.2.1	<b>Cenário 1 - Modelo + Visão</b> . . . . .	92
6.2.2	<b>Cenário 2 - Modelo + Encoders</b> . . . . .	95
6.2.3	<b>Cenário 3 - Modelo + IMU</b> . . . . .	97
6.2.4	<b>Cenário 4 - IMU + Visão</b> . . . . .	99
6.2.5	<b>Cenário 5 - IMU + Encoders</b> . . . . .	101

<b>6.2.6</b>	<b>Cenário 6 - Encoders + Visão . . . . .</b>	101
<b>6.2.7</b>	<b>Cenário 7 - Encoders + IMU . . . . .</b>	103
6.3	TESTE COM TRAJETO QUADRADO MENOR . . . . .	103
6.4	TESTE COM TRAJETO QUADRADO MAIOR . . . . .	107
6.5	CONSIDERAÇÕES FINAIS . . . . .	111
<b>7</b>	<b>CONCLUSÃO . . . . .</b>	115
	<b>REFERÊNCIAS . . . . .</b>	117
	<b>ANEXO A – Artigo publicado no simpósio CROS 2025 . . . . .</b>	125

## 1 INTRODUÇÃO

A utilização de robôs móveis cresce cada vez mais em diferentes áreas da sociedade, como medicina, agricultura, serviços e indústrias de diferentes tipos, como química, automotiva, metalúrgica, alimentícia, entre outras. De acordo com Statista (2023), a indústria global de robótica ultrapassou 37 bilhões de dólares e, se mantida a projeção de crescimento de 3,8% ao ano, até o final de 2028 alcançará o valor de 45 bilhões de dólares, mostrando o crescimento da utilização dos robôs em diversos setores do mercado.

Apesar de os robôs estarem sendo amplamente utilizados na indústria, eles também podem ser encontrados no dia-a-dia do ser humano cada vez mais, como robôs que limpam o chão de casa, que estão cada vez mais acessíveis. Além disso, pode-se citar o desenvolvimento e pesquisa de outras categorias de robôs, como sistemas de entrega por meio de drones (BENARBIA; KYAMAKYA, 2022) e carros autônomos (PAREKH et al., 2022).

A característica que difere os robôs móveis dos robôs industriais, como os robôs manipuladores utilizados para montagem de carros na indústria automotiva, é a capacidade de navegação, que acaba sendo um problema desafiador para os robôs móveis. A navegação pode ser dividida em 4 subsistemas basicamente, que são: mapeamento, localização, planejamento e desvio de obstáculos (RAJ; KOS, 2022).

No caso desse projeto, a questão principal é a localização, que é responsável por determinar a posição e orientação do robô no ambiente. Dentro do escopo da localização, os principais pontos são o posicionamento/localização global e o rastreio da posição de robôs móveis dentro de um mapa conhecido.

De acordo com Panigrahi e Bisoy (2022), o objetivo do rastreio da posição é acompanhar o posicionamento do robô a cada instância de tempo sabendo sua posição inicial, o que é possível por monitorar continuamente a rota do robô, seja por meio de sensores ou das equações cinemáticas que descrevem o robô. Já na localização global, a localização inicial não é conhecida e, assim, o robô deve se localizar dentro do ambiente global.

Basicamente, também segundo Panigrahi e Bisoy (2022), a localização pode ser dividida em duas etapas: a predição e a percepção. Na etapa de predição o robô faz o rastreio da posição utilizando sensores proprioceptivos, que medem informações do próprio robô e são atualizados numa alta frequência geralmente, para estimar sua posição. Entretanto, por conta do aumento da incerteza ao longo do tempo desse tipo de sensores, para a localização global é necessário

que o robô corria na etapa de percepção utilizando seus sensores exteroceptivos, que medem informações do robô em relação ao ambiente e são atualizados numa frequência baixa.

Para unir os dados desses diferentes sensores é utilizada uma técnica conhecida como 'fusão de sensores', cuja ideia geral é unir dados de diferentes sensores levando a uma análise mais profunda e complexa de uma situação, o que não seria possível utilizando esses dados separadamente e/ou de maneira singular (MENDES JR. et al., 2016). Assim, ao realizar a fusão dos dados de diferentes sensores, os pontos negativos de um sensor são minimizados por haver um outro sensor atuando nesse ponto fraco.

Um ambiente que possui a característica de um sensor externo com uma taxa de latência alta, onde os robôs precisam se posicionar com uma ótima precisão e navegam de maneira rápida e dinâmica dentro do ambiente, é o futebol de robôs da categoria *Small Size League* (SSL) da *RoboCup*. Por conta disso, a escolha dos sensores a serem utilizados, análises, testes e conclusões estão relacionadas com este ambiente. Basicamente, é importante que os robôs tenham um alto controle do seu posicionamento para que possam realizar jogadas e que não causem colisões com outros robôs, além de que na liga há um sistema central de visão por meio de câmeras que rodam a 60 frames por segundo (fps).

Embora a taxa de atualização do sistema de câmeras global da liga tenha uma taxa de atualização de aproximadamente 60fps, o que aparenta ser suficiente, a dinamicidade em que os jogos ocorrem na SSL, com a bola podendo chegar a uma velocidade de  $6.5m/s$ , implica numa alta necessidade de que as habilidades do robô estejam embarcadas no seu processador, tais como a navegação, controle de posição e, por conta disso, a localização. Assim, com a alta taxa de atualização do processador, os robôs têm acesso às informações dos seus diversos sensores de maneira quase instantânea, o que aumenta a performance deles durante uma partida (CHURCHLEY et al., 2015).

## 1.1 OBJETIVO

Este trabalho tem como objetivo realizar uma análise comparativa do uso de diferentes sensores para resolver o problema de localização e posicionamento de robôs móveis. No caso, os sensores a serem comparados foram a *Inertial Measurement Unit* (IMU), que é composta por um giroscópio e um acelerômetro, *encoders* acoplados às rodas do robô, o sistema de câmeras utilizado na categoria SSL e, por fim, o modelo cinemático de um robô omnidirecional. Com

essas combinações é possível verificar as diferenças da utilização desses sensores para um sistema de localização.

## 1.2 JUSTIFICATIVA

De acordo com Alatise e Hancke (2020), a navegação é um problema fundamental para a robótica que acaba dependendo de outros aspectos para o seu bom funcionamento, como a localização. A tarefa de localização de robôs móveis em um ambiente arbitrário é um desafio por conta da complexidade e diversidade de ambientes, métodos e sensores envolvidos. Além disso, os problemas de localização e navegação são o que acabam limitando a performance de robôs ainda.

Além disso, analisando trabalhos desenvolvidos na Small Size League que citam a questão da fusão de sensores (RYLL et al., 2013b; BAIE et al., 2017; ZOLANVARI et al., 2015; BEHZAD et al., 2019; NAEINI et al., 2020; CHURCHLEY et al., 2015; SALEHI et al., 2023), é possível notar a falta de trabalhos que façam as devidas comparações entre os diferentes sensores usualmente utilizados num robô da SSL, além de trabalhos que comparem os resultados obtidos por um sistema de localização baseado em fusão de sensores com o sistema global de câmeras da categoria.

Portanto, é importante que os diferentes sensores amplamente utilizados na categoria SSL para localização dos robôs e estimativa de posição sejam testados em um ambiente dinâmico, a fim de determinar os pontos fracos de cada um, formas de superar esses déficits e garantir o funcionamento deles de forma adequada em outros ambientes posteriormente.

## 1.3 ESTRUTURA DA DISSERTAÇÃO

No Capítulo 2 é feito o estudo do referencial teórico do futebol de robôs, onde os robôs da categoria SSL são descritos, os sensores a serem utilizados no trabalho e como estes devem ser calibrados. Já no Capítulo 3 são trazidos os conceitos necessários para entendimento do problema da localização de robôs móveis, com a explicação do método de fusão de sensores, além dos algoritmos para realização do método. No Capítulo 4 são apresentados os trabalhos relacionados, onde foram analisados os trabalhos correlatos que buscam resolver o problema da localização de robôs móveis, ou a estimativa de posição, mas também guiaram a escolha dos sensores, por exemplo, desse trabalho. No Capítulo 5 é apresentada a metodologia deste trabalho, onde estão descritos os testes, o domínio onde os testes foram realizados e como

foram avaliados os dados retirados dos testes. O Capítulo 6 traz os resultados obtidos no projeto, desde a implementação do sistema de *ground truth* desenvolvido, assim como a implementação dos cenários de fusão de sensores e os dados obtidos nos dois cenários de testes realizados. Por fim, o Capítulo 7 traz a conclusão do projeto e os possíveis trabalhos futuros.

## 1.4 CONTRIBUIÇÕES

Nesta seção serão destacadas as contribuições desta dissertação.

### 1.4.1 Publicações

Os resultados obtidos para obtenção do ground truth do robô durante os testes foram publicados no 1º Congresso Brasileiro de Robótica. No artigo foram apresentados os conceitos utilizados para o desenvolvimento do sistema, assim como os resultados obtidos comparado com o sistema de visão utilizado na categoria SSL.

A publicação também contou com uma apresentação de 15 minutos no simpósio.

### 1.4.2 Materiais eletrônicos

O repositório com os códigos do firmware<sup>1</sup> utilizado ao longo dos testes, assim como do sistema de *ground truth*<sup>2</sup> desenvolvido para esse projeto, foram disponibilizados para análise do desenvolvimento do projeto e para consulta em possíveis trabalhos futuros.

---

<sup>1</sup>[github.com/Joaovictor0508/object\\_detetection](https://github.com/Joaovictor0508/object_detetection)

<sup>2</sup>[github.com/Joaovictor0508/firmware\\_ssl](https://github.com/Joaovictor0508/firmware_ssl)

## 2 FUTEBOL DE ROBÔS

A ideia de robôs que jogam futebol foi proposta pela primeira vez pelo professor Alan Mackworth, em seu artigo *On Seeing Robots* (MACKWORTH, 1982). Independentemente, em outubro de 1992, um grupo de pesquisadores japoneses organizou um workshop sobre os grandes desafios em IA, onde iniciaram-se as primeiras discussões sobre usar o futebol para promoção da ciência e tecnologia. Em junho de 1993 foi organizada uma competição de robótica e, em menos de um mês, pesquisadores de fora do Japão começaram a pedir que essa iniciativa fosse ampliada para um projeto conjunto internacional. Com isso foi criada a RoboCup em 1997 (ROBOCUP, 2020)

A *RoboCup* busca promover pesquisas na área de robótica e inteligência artificial com um objetivo final de vencer a seleção campeã do mundo em 2050 com uma equipe totalmente autônoma de robôs humanoides (ROBOCUP, 2020).

O atual cenário competitivo da *RoboCup* mostra equipes tanto do ensino superior quanto do ensino básico que disputam diversas categorias, em eventos tanto a nível nacional quanto a nível internacional, tais como *RoboCup Soccer*, *RoboCup Rescue*, *RoboCup@home* e a *RoboCup Junior*.

### 2.1 SMALL SIZE LEAGUE

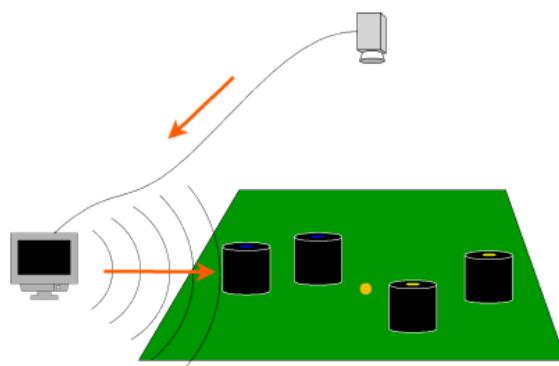
A *Small Size League* (SSL) é uma das ligas mais antigas da *RoboCup Soccer*, tendo o foco em solucionar o problema da cooperação e controle de robôs inteligentes num ambiente altamente dinâmico com um sistema híbrido centralizado/distribuído. A partida ocorre entre duas equipes utilizando seis ou onze robôs totalmente autônomos, que tem um limite de diâmetro e altura, e algumas outras restrições (ROBOCUP, 2020). Além disso, os robôs da liga são omnidirecionais, o que proporciona jogos dinâmicos com jogadas imprevisíveis.

Os jogos desta categoria são destacados por conta da alta velocidade tanto dos robôs, que podem chegar até 4m/s, quanto da bola utilizada, que pode chegar até 6.5m/s, mas também pela quantidade de robôs numa partida, que acontece entre dois times que podem ter de 6 a 11 robôs num campo de 9m X 6m a 12m X 9m, dependendo da divisão da partida (ROBOCUP-SSL, 2019b).

O tamanho do campo em relação aos robôs, a quantidade de robôs em uma partida e as velocidades dos robôs e da bolinha são algumas características que tornam a SSL um ótimo lugar para testes de algoritmos de localização e posicionamento.

Para realização da partida, um *setup* específico é necessário. No caso, acima do campo são instaladas câmeras, as imagens delas são processadas em um computador central que disponibiliza, a partir de pacotes de rede, as posições  $x$  e  $y$  e a orientação  $\theta$  dos robôs em campo, além das posições  $x$  e  $y$  da bola, como ilustrado na Figura 1. O processamento das imagens e cálculo das posições são realizados por um sistema desenvolvido pela própria liga, chamado de SSL-Vision (ZICKLER et al., 2010)

Figura 1 – Ilustração de uma partida da SSL.



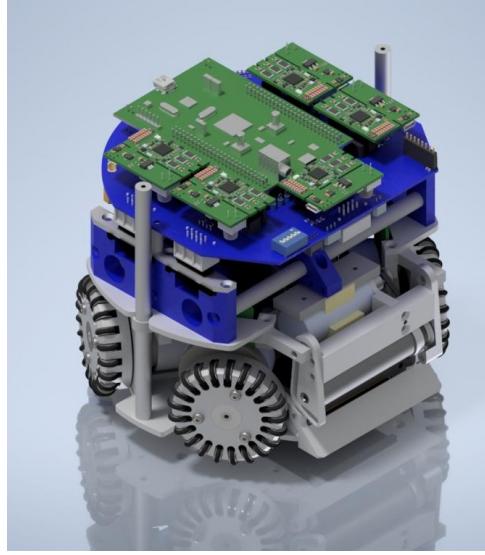
Fonte: Retirado de RoboCup-SSL (2019a)

Além disso, os robôs possuem limitações de tamanho, i.e., eles devem caber num diâmetro de 180mm e possuir uma altura máxima de 150mm (ROBOCUP-SSL, 2019b). A fim de lidar com essa limitação de dimensões e possuir robôs ágeis, as equipes utilizam um sistema de deslocamento omnidirecional, que é conseguido utilizando uma adaptação de rodas mecanum, em que os roletes são montados com uma certa angulação em relação ao eixo da roda (AGUIAR, J. V. L. et al., 2022).

Com esse sistema de deslocamento omnidirecional, o robô torna-se um sistema holonômico, i.e., o robô possui controle sobre todos os graus de liberdade da sua movimentação, ou seja, a rotação dele não interfere na translação, o que torna a SSL uma liga muito dinâmica e imprevisível. A Figura 2 mostra um robô SSL da equipe RoboFEI, nela é possível observar a roda omnidirecional utilizada.

No caso, a Equação (1) descreve um robô da categoria SSL, isto é, como teoricamente a posição de um robô propaga ao longo do tempo a partir de comandos de velocidade. As variáveis  $x_k$ ,  $y_k$  e  $\theta_k$  representam as posições linear e angular do robô no instante atual  $k$ , enquanto  $Vx_{k-1}$ ,  $Vy_{k-1}$  e  $\omega_{k-1}$  representam as velocidades linear e angular enviadas ao robô no instante de tempo anterior.

Figura 2 – Imagem de um robô SSL da equipe RoboFEI.



Fonte: Autor

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + \begin{bmatrix} dt & 0 & 0 \\ 0 & dt & 0 \\ 0 & 0 & dt \end{bmatrix} \times \begin{bmatrix} Vx_{k-1} \\ Vy_{k-1} \\ \omega_{k-1} \end{bmatrix} \quad (1)$$

Já o sistema de visão global da categoria SSL, que faz a captura de todos os robôs em campo e da bola, pode ser caracterizado conforme a Equação (2), em que  $\mathbf{z}_k$  representa o vetor de medições do SSL-Vision no instante  $k$ . É possível notar que a visão fornece diretamente a posição do robô acrescida de um certo ruído  $\sigma$ .

$$\mathbf{z}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \pm \begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_\theta \end{bmatrix} \quad (2)$$

### 2.1.1 Sistema de visão da categoria SSL

Durante a partida o processamento da tomada de decisões é feito num computador central de cada equipe, analisando as posições dos robôs e da bolinha em campo e, assim, enviando por meio de rádio frequência o que cada robô deve realizar, tal como: a velocidade de cada robô, se o robô deve chutar, se o robô deve ligar o dispositivo de drible, entre outros. Atualmente, na equipe RoboFEI, o cálculo do controle de posicionamento dos robôs é feito junto ao código da equipe e

somente a velocidade angular e linear que o robô deve impor é passado para ele no pacote de dados via rádio.

Apesar do sistema de câmeras ser suficiente para o jogo, há problemas com o controle de posição dos robôs por conta da latência consideravelmente alta da atualização de seus posicionamentos. Segundo Huang et al. (2020), a câmera envia imagens a cada 15ms, mas por conta da filtragem realizada pelo sistema a atualização da imagem pode demorar de 3 a 4 ciclos (40 a 60ms), portanto há uma considerável demora para que o pacote atual com os dados seja recebido pelas equipes, o que influencia diretamente na realização de jogadas.

O problema de alta latência de envio das imagens compromete a realização de jogadas durante as partidas e gera dificuldades no controle de posicionamento dos robôs. Assim, nota-se uma necessidade de aplicação de outras tecnologias para resolver esse problema de posicionamento, como utilização de outros sensores embarcados nos próprios robôs.

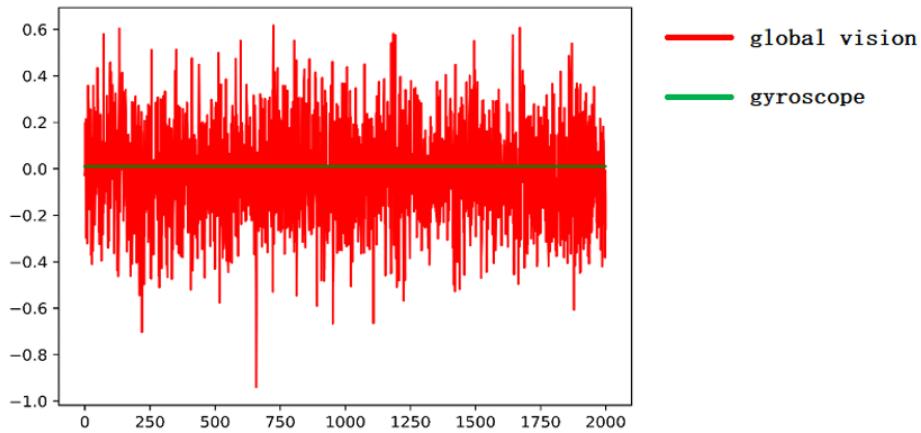
Segundo Huang et al. (2020), há quatro grandes problemas com o sistema de visão global da categoria. Em primeiro lugar, como dito, é o problema de que a frequência de atualização de 60Hz não é o suficiente para um controle de movimentação rápido e acurado. Segundo, a informação das posições que é enviada para as equipes possui uma quantidade alta de ruído, o que compromete altamente o controle de orientação dos robôs. Terceiro, a informação enviada aos times é previamente processada, levando de 3 a 4 frames (40-60ms) entre coletar a informação original da visão até obter a informação da visão. Quarto, a taxa de quadros do sistema de visão é instável, o que pode gerar perda de frames e, consequentemente, torna a frequência do controle instável.

A Figura 3 mostra uma comparação das informações de *feedback* da velocidade angular de um robô. É possível notar um ruído muito alto da informação vinda da visão global, enquanto em relação ao giroscópio mal é possível notar algum ruido. Isso mostra o quanto necessário é ter um sistema de localização que não seja totalmente dependente do sistema de câmeras da categoria.

A equipe RoboTeam Twente em seu *Team Description Paper* (TDP) de 2018 estima que o delay entre enviar um comando para o robô e notar uma resposta nas medições pode levar entre 80 e 150ms, dependendo da câmera utilizada (DOORNKAMP et al., 2018). Por conta disso, faz-se necessário um sistema de localização que possua um delay relativamente menor para que o controle de posição dos robôs seja feito adequadamente.

É notável que o sistema de câmeras da categoria SSL facilita a aquisição dos dados de posição, visto que ele já entrega para os times o posicionamento de todos os robôs e da bolinha.

Figura 3 – Comparação da velocidade angular adquirida pelo sistema de visão global e pelo giroscópio.



Fonte: Retirado de Huang et al. (2020)

Apesar disso, para realizar o controle dos robôs de maneira robusta e qualificada é necessária a utilização de um sistema de localização embarcado que realize estimativas de posicionamento por conta do tempo de atualização dos dados das câmeras ser relativamente alto para a dinamicidade da partida.

### 2.1.2 Fusão de sensores na *Small Size League*

A fusão de sensores aplicada na categoria SSL, principalmente para localização dos robôs em campo numa partida, é justificada pela dinamicidade em que as jogadas acontecem por conta da velocidade máxima imposta na bola de jogo para realização de passes e chutes (ROBOCUP-SSL, 2019b), além do problema já abordado da taxa de atualização das posições capturadas pelo sistema global de câmeras.

Por conta da complexidade de trabalhar com diversas habilidades, como localização, navegação e controle, de forma embarcada, grande parte das equipes prefere executar o controle de posicionamento no computador central de estratégia, utilizando somente os dados adquiridos do sistema global de visão, e enviando para os robôs, via rádio, a velocidade local que cada um deles deve impor (CHURCHLEY et al., 2015; SALEHI et al., 2023). Assim, de forma embarcada os robôs precisam somente de um feedback da rodas por meio de encoders, por exemplo, para manter um controle de velocidade delas.

Apesar de ser um sistema mais simples de implementar, o controle de posição dos robôs no computador central acaba levando a uma menor performance comparado com a utilização do controle embarcado (CHURCHLEY et al., 2015), muito por conta do delay inerente para atualização das imagens do sistema global de câmeras da liga. Entre um frame e outro, por conta da alta velocidade da bola em jogo, somado com o delay de envio e recebimento dos dados via rádio-frequência, há um aumento de dificuldade para realização de certas jogadas e habilidades (SALEHI et al., 2023).

Para solucionar o problema do delay entre pacotes do sistema de visão global da liga, a fusão de sensores passou a ser utilizada por equipes que, nos anos seguintes, passaram a demonstrar uma qualidade de jogo superior (CHEN et al., 2018; RYLL et al., 2013a; CHURCHLEY et al., 2015; BAIE et al., 2017; SALEHI et al., 2023; ZOLANVARI et al., 2015), a fim de aumentar a realização de habilidades de forma embarcada no robô. Portanto, um sistema de localização preciso e com baixa taxa de atualização é necessário, utilizando dados mais confiáveis na malha de controle.

Em Salehi et al. (2023) é feita uma fusão de sensores para orientação entre os dados do sistema de visão e os dados da IMU, melhorando a qualidade da orientação predita. Em Behzad et al. (2019) a fusão de sensores também é feita especialmente para a orientação, sendo que a equipe coloca que comparado com o SSL-Vision a fusão de sensores é mais acurada.

A partir disso, é notória a melhoria de performance levando as habilidades do robô, como localização e navegação, para o microcontrolador do robô. A informação dos sensores presentes nos robôs e alta taxa de processamento permitem que o robô tenha melhores estimativas de localização em campo entre um frame e outro do sistema global de câmeras, aumentando a performance dos robôs.

## 2.2 SENSORES

A utilização de sensores é parte essencial para o funcionamento correto de um robô. Segundo Oliveira et al. (2017), o mais predominante em robôs industriais são robôs projetados para realizarem operações pré-programadas, que acabam não usufruindo de sensores para atingirem seu objetivo. Entretanto, para robôs mais complexos, os sensores acabam introduzindo um maior nível de inteligência para poder interagir com o meio que está inserido por meio de atuadores.

Também de acordo com Oliveira et al. (2017), a utilização de sistemas sensoriais faz com que robôs sejam mais facilmente adaptáveis a uma maior gama de tarefas, atingindo um maior grau de universalidade, diferente dos robôs pré-programados, que acabam realizando apenas uma única função. Um robô que, a partir da leitura dos sensores, possui sensações tal como um humano, é mais facilmente treinado para realização de tarefas complexas.

Portanto, essa seção visa apresentar as diferentes categorias de sensores, tal como explicar o funcionamento geral dos sensores a serem utilizados neste trabalho e como eles devem ser calibrados para conseguir aproveitar o melhor dos dados apresentados por eles. Os sensores descritos nessa seção são os usualmente utilizados pelos robôs da SSL, por conta da configuração do sistema global de visão e do limite de tamanho descrito em regra.

### **2.2.1 Tipos de sensores**

De acordo com Sasiadek e Hartana (2000), os sensores podem ser divididos em duas categorias principais: internos e externos. Essa diferenciação diz respeito a partir de onde vem a informação lida pelo sensor, ou seja, se é do mundo externo ou se é internamente do próprio robô.

Sensores internos fornecem informações sobre parâmetros internos do robô, ou seja, medem variáveis físicas dele, como a velocidade e o sentido de rotação de um motor, ou o ângulo de uma junta, como exemplos. Alguns possíveis sensores que fazem parte desse tipo são: encoder, giroscópio, acelerômetro, bússolas.

Já os sensores externos medem a relação entre o robô e o ambiente em que ele está inserido, que podem ser objetos naturais ou artificiais, como por exemplo a distância do robô até um objeto ou medidas químicas do ambiente. Alguns possíveis sensores que fazem parte desse tipo são: sensores de contato (bumpers), sensores de distância como laser, sonar e radar, entre outros.

Ambos sensores possuem vantagens e desvantagens. Para períodos curtos de tempo, as medições de sensores internos são bem acuradas, embora a longo prazo as medidas normalmente passam a ter desvios e erros. Ao contrário disso, os sensores externos não tem problemas de desvio do sinal ao passar do tempo, mas as medidas deles normalmente não estão sempre disponíveis, ou seja, possuem um período grande para atualizarem suas medidas.

Portanto, para obter resultados ótimos, ambos sensores são normalmente combinados, juntando as qualidades de ambos e fazendo com que as desvantagens deles sejam superadas.

Por conta do erro de ambos os sensores, é realizada uma fusão das medidas dos dois tipos de sensores, o que irá produzir uma estimativa desejada da posição do robô.

No caso deste projeto, os seguintes sensores foram utilizados e explicados de maneira mais aprofundada: câmera, encoder, giroscópio e acelerômetro.

### **2.2.2 Sistema de câmeras**

A câmera é um instrumento cujo uso em aplicações na área da robótica tem crescido bastante. Mapeamento, localização, navegação, desvio de obstáculos, reconhecimento de objetos e inspeção de qualidade são só alguns exemplos de possíveis utilizações de câmeras para realização de tarefas por parte de robôs. No centro dessa ascensão das câmeras está a evolução tanto dos processadores quanto dos algoritmos de visão computacional avançados.

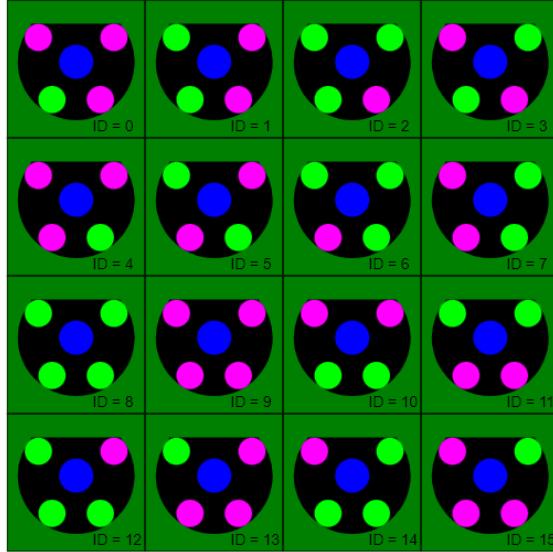
Segundo TechNexion (2023), câmeras são cruciais no campo da robótica guiada por visão por aperfeiçoar as habilidades de percepção. Um robô pode aprender muito sobre seu arredor a partir dos dados visuais que câmeras coletam. Robôs podem obter informações valiosas desses dados utilizando diferentes métodos de processamento, permitindo que o robô enxergue, comprehenda e interaja com o ambiente de uma maneira mais profunda.

No caso deste projeto, a utilização de câmeras se dá externamente aos robôs dentro da categoria SSL da RoboCup, já que o sistema de visão não é embarcado nos robôs, embora haja estudos para alocar uma câmera dentro dos robôs da categoria de pequeno porte, como trazido por Melo e Barros (2022).

Detalhando um pouco mais o sistema de visão por câmeras da categoria SSL da RoboCup, atualmente utiliza-se uma ou duas câmeras, dependendo se o campo é da divisão A ou da divisão B. Independente do caso, a câmera fica posicionada acima do campo a 6 metros de altura e fica conectada a um computador central. Nele, a imagem é recebida, tratada e processada, identificando a posição da bola e dos robôs a partir do padrão de cor posicionado na parte de cima dos robôs, como é possível observar na Imagem 4 as 16 diferentes combinações possíveis (ZICKLER et al., 2010).

Tanto o computador onde as imagens são processadas quanto o computador de cada uma das equipes estão conectadas numa mesma rede. Assim, após o processamento das imagens, as posições dos robôs e da bola são passadas para as equipes pela rede, por isso que o sistema de visão é dito compartilhado, pois ambas as equipes recebem as mesmas informações.

Figura 4 – Padrões de cores para identificação dos robôs.



Fonte: RoboCup-SSL (2019b)

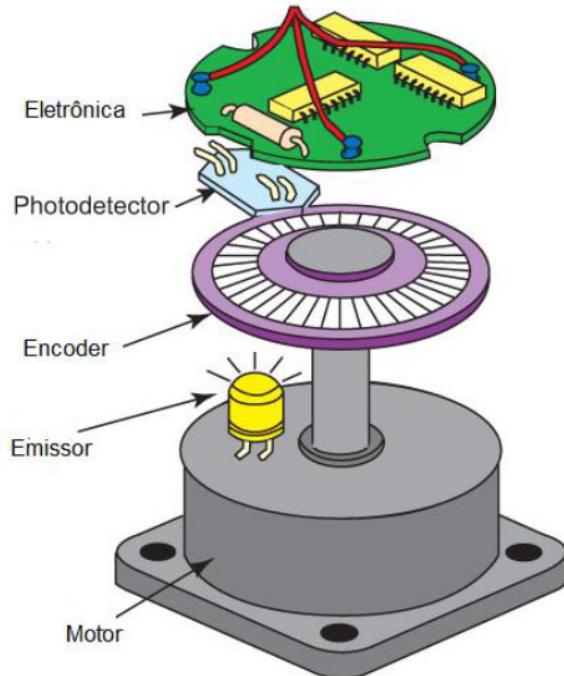
### 2.2.3 Encoder

Encoders são dispositivos utilizados a fim de medir o estado interno e a dinâmica de um robô móvel. Eles possuem uma vasta gama de aplicações fora da robótica e, por conta disso, robôs se aproveitaram dos benefícios da alta qualidade e baixo custo de sensores que oferecem uma excelente resolução de leitura. No mercado existem alguns diferentes tipos de encoders, como os ópticos e magnéticos.

No caso deste projeto, é utilizado um encoder do tipo óptico. Segundo Siegwart, Nourbakhsh e Scaramuzza (2011), este tipo de encoder se tornou o dispositivo mais popular para medição de velocidade e posição angulares de um motor, do eixo de uma roda ou mecanismo de direção.

Um encoder óptico é basicamente um picador de luz mecânico que produz uma certa quantidade de pulsos na forma de um seno ou quadrado para cada revolução. No caso, o sensor consiste de uma fonte de iluminação, uma "grade" fixa que mascara a luz, um disco rotor com uma grade óptica fina que gira com o eixo e um detector óptico fixo. Ao passo que o rotor se movimenta, a quantidade de luz atingindo o detector óptico varia baseado no alinhamento das grades fixas e móveis. É possível observar a montagem e ter uma melhor ideia do funcionamento de um encoder óptico com a Figura 5.

Figura 5 – Ilustração da montagem e funcionamento de um encoder óptico.



Fonte: Adaptado de Luiz R. (2021)

O projeto RoboFEI utiliza motores brushless da empresa Maxon®, no caso um motor brushless EC 45 com 50W acoplado em cada uma das rodas (MAXON GROUP, 2019). Já o encoder utilizado é da empresa US Digital®(US DIGITAL, 2024), no caso é utilizado um encoder do modelo E4T por roda também, sendo que eles ficam acoplados diretamente ao eixo do motor.

Cada encoder está ligado a um timer do microcontrolador utilizado em cada robô para realizar a contagem de pulsos do sensor. Seguindo a Equação (3), pode-se determinar a velocidade de rotação de cada roda em rpm, conhecendo a contagem de pulsos realizada, o intervalo de tempo (em segundos), a transmissão envolvida no conjunto roda-motor e a resolução de leitura do encoder (quantidade de pulsos por volta).

$$\text{Rotação} = \frac{\text{Pulsos}}{\text{Transmissão} \times \text{Pulsos por volta}}$$

$$\text{Velocidade} = \frac{\text{Rotação} \times 60}{\Delta t} \quad (3)$$

## 2.2.4 Giroscópio

Giroscópios também são um dos principais sensores utilizados em robôs para realização de tarefas básicas como navegação. De acordo com Dingman (2020), eles são componentes essenciais de sistemas complexos utilizados em todas aplicações aeroespaciais, industriais e na área da robótica. Giroscópios auxiliam desde aviões e barcos até drones e carros autônomos a navegarem.

Segundo Passaro et al. (2017), giroscópios são dispositivos montados em uma estrutura e capazes de detectar uma velocidade angular se a estrutura estiver girando. Esse sensor pode ser utilizado de forma sozinha ou pode estar incluso em um sistema mais complexo, como uma bússola giroscópica, uma IMU (*Inertial Measurement System*, do inglês) ou um INS (*Inertial Navigation System*, do inglês), por exemplo.

No livro '*Introduction to Autonomous Mobile Robot*', os autores trazem que giroscópios são sensores de direção que preservam sua orientação em relação a um *frame* de referência fixo (SIEGWART; NOURBAKHS; SCARAMUZZA, 2011). Por isso, eles fornecem uma medida absoluta de orientação de um sistema móvel.

Também de acordo com Siegwart, Nourbakhsh e Scaramuzza (2011), os giroscópios são divididos em duas categorias: mecânicos e ópticos. Os primeiros dependem das propriedades de um rotor de rotação rápida, propriedade chamada de precessão giroscópica. Já os segundos são sensores de velocidade angular que utilizam dois feixes de luz monocromáticos, ou lasers, emitidos de uma mesma fonte

No caso desse projeto, o giroscópio utilizado é o I3G4250D, que é um sensor de velocidade angular de baixo consumo de energia capaz de realizar medidas nos 3 eixos (STMICROELECTRONICS, 2015). Esse componente inclui o sensor e uma interface capaz de fornecer a medida de velocidade angular ao mundo externo por meio de uma interface digital I<sup>2</sup>C (do inglês, *Inter-Integrated Circuit*) ou SPI (do inglês, *Serial Peripheral Interface*).

## 2.2.5 Acelerômetro

O acelerômetro é mais um dos sensores que é utilizado para que o robô possua a habilidade de entender sua localização no espaço, o que é criticamente importante para alcançar com êxito o objetivo determinado para o robô desenvolvido.

Segundo Nistler e Seleka (2011), grande parte dos sistemas de odometria para aplicações em robótica possuem acelerômetros. Estes continuamente medem a aceleração do robô, que

é integrada para determinada a velocidade dele, e é integrado novamente para ter a medida da posição relativa ao ponto inicial.

Entretanto, por conta da influência da gravidade, da força inercial de Coriolis (PERSSON, 1998) e componentes rotacionais de aceleração, sistemas de odometria baseados em acelerômetros estão sujeitos a diversos erros dependendo do processamento das medidas do sensor. Também segundo Nistler e Selekwa (2011), quando o robô se move numa superfície horizontal, a velocidade computada irá refletir a velocidade real do robô, mas em superfícies inclinadas, a velocidade medida irá incluir esses erros listados, que não fazem parte da velocidade real.

De acordo com Dadafshar (2014), a operação básica de um acelerômetro recai na Segunda Lei de Newton, a qual diz que a aceleração de um corpo é diretamente proporcional, e na mesma direção, a força resultante atuante no corpo, e inversamente proporcional à massa do corpo, descrito na Equação (4).

$$\vec{a} (m/s^2) = \frac{\vec{F}(N)}{m(kg)} \quad (4)$$

Nota-se que a aceleração gera uma força que é capturada pelo mecanismo de detecção de força do acelerômetro. Portanto, o acelerômetro na verdade realiza medidas de força, e não aceleração, mas ele acaba medindo a aceleração indiretamente por meio da força aplicada em um de seus eixos.

De acordo com Siegwart, Nourbakhsh e Scaramuzza (2011), os acelerômetros são separados dependendo do princípio físico utilizado para realizar a medição da deflexão da massa interna do sensor. Um mecanismo comum de detecção utilizado é a detecção por capacitância, que mede a deflexão ao medir a capacidade entre uma estrutura física e a massa interna. Outra alternativa de medição é a piezoeletrica, que é baseada na propriedade de certos cristais em gerarem tensão quando um estresse mecânico é aplicado neles, no caso a massa interna é posicionada no cristal e, quando uma força externa é aplicada, a massa induz uma tensão que pode ser medida.

No caso desse projeto, o acelerômetro utilizado é o LSM303AGR, que é um sensor digital de aceleração linear capaz de realizar medidas nos 3 eixos, e é um sensor digital magnético nos 3 eixos também (STMICROELECTRONICS, 2022). O componente inclui uma interface serial I<sup>2</sup>C, que suporta os modos padrão e rápido com 100kHz e 400kHz, ou uma interface serial padrão SPI.

### **2.2.6 Inertial Measurement Unit (IMU)**

A IMU, é um dispositivo que utiliza giroscópios e acelerômetros para estimar a posição, velocidade e aceleração relativos do veículo em movimento. Este componente se tornou comum em aviões e barcos, por exemplo, por estimar a posição do veículo em seis graus de liberdade, no caso: posição(x, y, z) e orientação (*roll, pitch, yaw*) (SIEGWART; NOURBAKHS; SCARAMUZZA, 2011).

Além disso, as IMUs comercializadas também estimam velocidade e aceleração. Considerando que a IMU possua 3 acelerômetros ortogonais e 3 giroscópios ortogonais, os dados do segundo são integrados para estimar a orientação do veículo enquanto os dados do primeiro são utilizados para estimar a aceleração instantânea do veículo.

A aceleração é transformada para o frame da navegação local por meio da estimativa atual da orientação do veículo relativo à gravidade. Portanto, o vetor gravidade pode ser subtraído das medidas, resultando numa aceleração que é integrada para obter a velocidade e integrada novamente para obtenção da posição. Para sobrepor o problema da necessidade de conhecer a velocidade inicial, a integração é tipicamente iniciada no repouso, ou seja, velocidade igual a zero.

IMUs são extremamente sensíveis na questão de erros de medidas tanto em relação ao giroscópio quanto ao acelerômetro. Por exemplo, o desvio no giroscópio inevitavelmente prejudica a estimativa da orientação do veículo relativa à gravidade, o que resulta numa cancelamento incorreta do vetor da gravidade. Além disso, por exemplo, os dados do acelerômetro são integrados duplamente para obter a posição, logo qualquer resíduo do vetor gravidade gera um erro que é duplamente integrado na posição. Por conta desse problema de desvio, é necessário alguma referência de fonte externa de medida, como um GPS (do inglês, *Global Positioning System*), câmera ou LiDAR (do inglês, *Light Detection And Ranging*).

### **2.2.7 Calibração dos sensores**

O mercado mundial de sensores vem expandindo numa alta taxa ao longo dos últimos anos empurrado pelo desenvolvimento de outras tecnologias que fazem uso desses componentes, como robôs, carros autônomos, tecnologias de energia verde e internet das coisas (*Internet of Things*, do inglês), por exemplo. De acordo com BCC (2024), o mercado global de sensores estima o crescimento de \$179.7 bilhões em 2023 para \$300.5 bilhões até 2029.

Entretanto, apesar do forte avanço do mercado de sensores mundial, de acordo com Jain, Zhang e Jiang (2019), a calibração de sensores é um dos tópicos menos discutidos no desenvolvimento de sistemas autônomos, apesar de ser o bloco de fundação do sistema e de seus sensores, e é uma etapa de processamento necessária antes da implementação de técnicas de fusão sensorial.

De acordo com Lv et al. (2020), a calibração dos sensores é uma parte fundamental para o desenvolvimento de um projeto de fusão multi-sensorial. Isso se dá pelo fato do aumento da qualidade dos dados lidos pelos sensores e, assim, uma consequente melhoria na confiabilidade do sistema como um todo. Em sistemas como robôs e carros autônomos, isso pode determinar diretamente a segurança e viabilidade deles.

No caso desse projeto, a calibração foi realizada para os sensores giroscópio e acelerômetro, que são chamados de sensores inerciais microeletromecânicos (MEMS, do inglês), cujo desenvolvimento foi o protagonista para o crescimento de sistemas de navegação inerciais (INS, do inglês) e superar os pontos negativos de outros sensores, como o GPS, que são consideravelmente lentos para atualizar a informação.

Segundo Papafotis e Sotiriadis (2020), sensores inerciais MEMS são utilizados preferencialmente por conta tanto do seu baixo custo quanto do seu reduzido tamanho. Entretanto, uma grande desvantagem deles é a sua característica de grande erro. Por conta disso, a calibração desses sensores é necessária para garantir seu bom funcionamento num INS, compensando a parte determinística de seu erro.

#### **2.2.7.1 Calibração do acelerômetro**

A calibração do componente acelerômetro pode ser dividida em duas partes diferentes: compensação do *bias* da leitura de cada um dos eixos e a calibração da inclinação da IMU em relação ao frame do robô, este que acontece por conta da solda do componente ou de algum fator mecânico, o que faz com que os eixos do robô e do componente provavelmente não sejam compatíveis.

A calibração é realizada pelo método proposto por Menezes Filho et al. (2020), que é baseado numa estimativa pelo método dos mínimos quadrados. No caso, o método é uma adaptação de uma técnica utilizada para calibração de magnetômetros (dispositivos que medem a força do campo magnético), alterando que ao invés de medir o campo magnético da Terra é medida a aceleração local da gravidade. Além disso, o método para acelerômetros deve ser

realizado enquanto o componente esteja estacionário ou submetido a rotações que não produzam forças centrípetas detectáveis.

De acordo com Menezes Filho et al. (2020), há 4 componentes que compõe o erro de um acelerômetro, que são:

- a) **Desvio:** Chamado de *bias* em inglês, o desvio é o componente mais comum na calibração de acelerômetro. Esse tipo de erro adiciona um deslocamento nas leituras do sensor.
- b) **Fatores de escala:** Esse componente corrompe as medições ao escalar incorretamente elas.
- c) **Desalinhamentos:** Também conhecido como não-ortogonalidades ou erros de acoplamento cruzado, esse componente diz respeito à disposição angular entre os eixos do corpo e os eixos reais de sensibilidade. O efeito prático desses desalinhamentos é que um sensor acaba sentindo (leia-se, medindo) a aceleração dos outros eixos, portanto a leitura não é completamente relacionada ao seu respectivo eixo.
- d) **Ruídos aleatórios:** Este componente é assumido como ruído branco e com média zero seguindo uma distribuição Gaussiana. Os autores colocam que como a aceleração é realizada no domínio dela própria nenhuma integração numérica é realizada, logo esses processos aleatórios podem ser desconsiderados.

De acordo com Hassan e Bao (2020), apenas os três primeiros itens são considerados durante a modelagem de erros sistemáticos. Assim, a leitura de um acelerômetro pode ser descrita conforme a Equação (5), em que  $a$  é o vetor que representa as acelerações calibradas nos 3 eixos,  $S$  é a matriz que representa os erros de fator de escala e desalinhamento (no caso, na diagonal principal estão os dados do erro de fator de escala, enquanto o restante dos valores são os erros de desalinhamento), o vetor  $B$  representa os desvios do sensor em cada eixo, indicados como  $b$ , enquanto  $\tilde{a}$  representa os dados crus do sensor nos 3 eixos. Os índices  $x$ ,  $y$  e  $z$  representam os eixos em questão e, quando eles estão conjugados (como  $xy$ , por exemplo) representam influência mútua.

$$\begin{aligned} a &= S(\tilde{a} - B) \\ \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} &= \begin{bmatrix} S_x & S_{xy} & S_{xz} \\ S_{xy} & S_y & S_{yz} \\ S_{xz} & S_{yz} & S_z \end{bmatrix} \left( \begin{bmatrix} \tilde{a}_x \\ \tilde{a}_y \\ \tilde{a}_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \right) \end{aligned} \quad (5)$$

Como dito anteriormente, o método se baseia que durante o repouso do sensor, o módulo da aceleração nos três eixos deve ser igual à aceleração da gravidade, como descrito na Equação (6).

$$a_x^2 + a_y^2 + a_z^2 = g^2 \quad (6)$$

A partir disso, devem ser recolhidas amostras do sensor em diferentes posições para que a calibração tenha uma maior eficiência, já que em Menezes Filho et al. (2020) é mostrado que o método consegue melhores resultados ao aumentar a quantidade de amostras, sendo que os autores colocam que 9 é a quantidade mínima nesse caso.

#### 2.2.7.2 Calibração do giroscópio

A calibração do giroscópio segue as mesmas ideias da calibração do acelerômetro. Entretanto, em Hassan e Bao (2020) somente o fator de escala e os desvios são considerados no estudo como fontes de erro.

Assim, a relação entre a velocidade angular real e a velocidade angular medida pode ser observada com a Equação (7), em que  $\omega$  é o vetor que representa as velocidades angulares calibradas nos 3 eixos, enquanto  $\tilde{\omega}$  representa o vetor com os dados crus do sensor nos 3 eixos.

$$\begin{aligned} \omega &= S(\tilde{\omega} - B) \\ \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} &= \begin{bmatrix} S_x & S_{xy} & S_{xz} \\ S_{xy} & S_y & S_{yz} \\ S_{xz} & S_{yz} & S_z \end{bmatrix} \left( \begin{bmatrix} \tilde{\omega}_x \\ \tilde{\omega}_y \\ \tilde{\omega}_z \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} \right) \end{aligned} \quad (7)$$

Entretanto, como nesse caso as amostras também devem ser tomadas em repouso, a aceleração real deve ser considerada nula. Portanto, os erros de fator de escala podem ser desconsiderados no caso desse projeto, sendo necessário apenas descobrir os desvios do sensor.

### 3 LOCALIZAÇÃO DE ROBÔS MÓVEIS

Nessa seção serão comentadas as questões que envolvem como um todo a localização de robôs móveis. No caso, serão discutidos a questão geral da localização de robôs e seus principais problemas, suas diferentes instâncias e a questão da informação disponível para localização de robôs.

#### 3.1 O PROBLEMA DA LOCALIZAÇÃO

A navegação é uma das competências mais desafiadoras necessárias em um projeto de robô móvel. De acordo com Siegwart, Nourbakhsh e Scaramuzza (2011), o sucesso da navegação depende do sucesso de 4 pilares principais: percepção, localização, cognição e controle de movimento. O primeiro é como o robô interpreta os dados dos sensores para extrair dados significativos para seus objetivos. O segundo é a determinação da posição do robô no ambiente. O terceiro é sobre como o robô deve decidir suas ações para atingir seus objetivos. O quarto define como o robô deve modelar as saídas dos seus motores para atingir a trajetória desejada.

Este trabalho se debruça no pilar da localização basicamente. O problema da localização consiste em responder a pergunta "Onde estou?" do ponto de vista do robô, o que quer dizer que o robô precisa descobrir sua localização relativa ao ambiente em que ele se encontra. Quando fala-se sobre posição, quer dizer sobre as coordenadas  $x$  e  $y$  do robô, tal como sua orientação no sistema de coordenadas global.

Como dito em Thrun et al. (2001), o problema de localização de um robô é algo muito importante, sendo um componente chave em diversos sistemas robóticos autônomos de sucesso. Se um robô não sabe onde está relativamente ao ambiente, a tomada de decisão do que fazer é praticamente impossível, o robô precisa ter pelo menos uma certa noção de onde ele está para poder operar e agir de maneira certa. Segundo Borenstein et al. (1997), saber exatamente a posição de um robô é um problema fundamental em aplicações de robôs móveis para prover realmente capacidades autônomas.

Problemas de localização são caracterizados pelo tipo de conhecimento que está disponível inicialmente e durante a execução. Basicamente, há três tipos de problemas de localização com diferentes graus de dificuldade, que são:

- a) **Rastreio de posição:** Assume que a posição inicial do robô é conhecida, logo a localização do robô pode ser conseguida ao acomodar o ruído na movimentação do

robô, geralmente o efeito desse ruído sendo pequeno. Esse problema é dito como local, já que a incerteza é local e restrita a uma região perto da posição verdadeira do robô, além de que a incerteza é geralmente aproximada por uma distribuição unimodal, como uma gaussiana.

- b) **Localização Global:** Aqui a posição inicial do robô é desconhecida, já que o robô é inicialmente colocado em algum local do ambiente, mas há a falta do conhecimento de onde é o local, ou seja, ele precisa se localizar do zero. As abordagens para localização global não podem assumir limite no erro da posição, assim como assumir distribuição probabilística unimodal é geralmente inapropriado.
- c) **Problema do sequestro de robô:** É uma variante do problema de localização global, mas nesse caso o robô sabe onde está localizado e de repente é "sequestrado" para outra localização sem que o robô esteja ciente disso. O problema é o robô detectar que foi sequestrado e, em seguida, descobrir sua nova localização. A importância prática disso, apesar de ser algo que não aconteça frequentemente, decorre que grande parte dos algoritmos de localização não garantem que nunca falharão.

### 3.2 OS DESAFIOS DA LOCALIZAÇÃO

Ao falar dos desafio da localização, Siegwart, Nourbakhsh e Scaramuzza (2011) falam sobre a situação hipotética de utilizar um GPS (do inglês, *Global Positioning System*) em um robô móvel e como o problema de localização estaria evitado, já que o sensor informaria a posição exata interna e externamente e a questão "Onde estou?" sempre estaria respondida. Porém, infelizmente, esse sensor não é prático, já que o GPS atual tem uma acurácia de alguns metros na época de publicação do livro, o que é inaceitável para localizar robôs móveis, além de não funcionar em ambientes internos. Atualmente, a acurácia já está em centímetros para aplicações externas, por exemplo.

Indo mais a fundo nas limitações do GPS, a localização é mais do que saber a posição absoluta do robô em relação à Terra, é também saber a sua localização relativa em respeito a, por exemplo, humanos, considerando a situação de um robô que tem que interagir com pessoas. Além do mais, se o robô planeja atingir uma localização específica, talvez seja necessário adquirir um modelo do ambiente (um mapa) e identificar a posição relativa do robô nele.

Por conta da falta de acurácia e imperfeição de sensores e atuadores que a localização enfrenta desafios difíceis. Os principais aspectos que tornam o funcionamento de sensores e atuadores sub-ótimos são: ruído e *aliasing* em sensores e ruído em atuadores.

Sensores são a entrada fundamental do robô para o processo de percepção e, portanto, o grau em que sensores podem discriminar o estado em relação ao mundo que o robô se encontra é crítico. O ruído induz uma limitação na consistência das leituras de um sensor em um mesmo estado do ambiente. Geralmente, a fonte de problemas com ruídos em sensores é que algumas características não são capturadas pelo robô e, portanto, ignoradas. Resumindo, o ruído em sensores reduzem a informação útil de suas leituras. Uma saída para isso é considerar várias leituras, aplicando fusão temporal ou fusão de diversos sensores para aumentar a qualidade geral da informação de entrada de robôs.

Outra deficiência em relação aos sensores é a questão do *aliasing*, que os leva a colherem pouco conteúdo informativo, o que acaba agravando o problema da percepção e, assim, dificultando a localização de robôs móveis. Um exemplo que mostra bem a questão do *aliasing*, trazido em Siegwart, Nourbakhsh e Scaramuzza (2011), é que a utilização de um sonar em um robô móvel não traz a informação se algo que foi detectado é um humano que o robô deveria dizer "com licença" ou um objeto inanimado que o computador deveria recalcular o trajeto para ultrapassar. Ou seja, a quantidade de informações é geralmente insuficiente para identificar a posição do robô a partir de uma leitura de percepção única.

Já o ruído em atuadores cai na questão de que uma única ação tomada por um robô móvel pode ter diferentes resultados possíveis, mesmo que da perspectiva do robô o estado inicial antes da ação tomada é bem conhecido. Em resumo, atuadores em robôs móveis introduzem incerteza sobre o estado futuro, como por exemplo, o ato de andar tende a aumentar a incerteza de um robô. A maior fonte de erro geralmente reside em um modelo incompleto do ambiente, como por exemplo o fato de o modelo não levar em conta que as rodas de um robô podem escorregar ou que um humano pode empurrar o robô, ou seja, não leva em conta possíveis fontes de erros que não podem ser modeladas, resultando numa falta de acurácia entre o movimento físico do robô, a movimentação pretendida pelo robô e a estimativa de movimentação pelo sensor.

### 3.3 INFORMAÇÃO DISPONÍVEL

Para determinar sua localização, um robô tem acesso a dois tipos de informação. Primeiro por meio de uma compreensão a priori obtida pelo próprio robô ou suprida por uma fonte externa

numa fase chamada de inicialização. Segundo o robô obtém informação sobre o ambiente por meio de cada observação e ação realizadas durante a fase chamada navegação.

Em geral, a informação a priori fornecida ao robô descreve o ambiente pelo qual o robô está navegando, ou seja, especifica algumas características que são variantes no tempo e assim podem ser utilizadas para determinar a localização. Alguns exemplos desse tipo de informação podem ser mapas e relações causa-efeito.

Robôs podem ter acesso a um mapa que descreve o ambiente em que está localizado. Os mapas podem ser topológicos ou geométricos (MENDES, 2017), o primeiro tipo descreve o ambiente em termos métricos, como por exemplo mapas de rodovias, já o segundo tipo descreve o ambiente em termos de características específicas em localizações e maneiras de ir de um local para outro. O mapa pode ser aprendido pelo robô previamente, ou fornecido por uma fonte externa, ou aprendido enquanto navega pelo ambiente. Já as relações causa-efeito fornecem informações a priori ao robô por meio de uma dada entrada de observação, dizendo ao robô onde ele está a partir delas.

Robôs também tem acesso à chamada informação de navegação, que é o tipo de informação que o robô reúne de diferentes sensores enquanto navega pelo ambiente. Um robô tipicamente performa dois tipos de ações ao navegar: ele anda ou age no ambiente por um lado, e sente o ambiente por outro lado.

Um sistema de locomoção consiste de rodas, pernas ou trilhos, ou qualquer coisa que faça o veículo se movimentar pelo ambiente. A maneira na qual um sistema de deslocamento muda a localização contém informação valorosa para realizar a estimativa da própria localização, ou seja, saber os efeitos de ações executadas pelo sistema indica diretamente a localização do veículo depois da execução dessas ações.

O robô sente o ambiente por meio de sensores, que indicam a informação de uma situação momentânea, chamada de observação ou medição, ou seja, essa informação descreve uma situação do ambiente do robô em um certo momento. Observações feitas do ambiente providenciam informação sobre a localização do robô que é independente de uma estimativa de localização anterior, dando ênfase que a informação dessas medições vem da observação do ambiente ao invés do próprio robô.

### 3.4 FUSÃO DE SENSORES

Alatise e Hancke (2020) trazem que robôs móveis autônomos estão se tornando mais proeminentes nos últimos tempos por conta do aumento de sua relevância e aplicações em diversas áreas, como em empresas, indústrias, hospitais, setor agrícola, realizando funções como carregamento de objetos pesados, monitoramento e busca. Por conta disso, a fusão de sensores vem sendo utilizada para solução de problemas como localização, mapeamento e navegação.

A fusão de sensores é um tema que envolve uma grande multidisciplinaridade de áreas. Por conta disso, existem diversas definições do que é fusão de sensores na literatura, como a definição de Castanedo et al. (2013) e Nagla, Uddin e Singh (2014), que definem como o uso cooperativo de informação providenciada por diversos sensores a fim de ajudar no desempenho de uma determinada função. Trazendo mais para o campo da robótica, Luo, Chang e Lai (2011) trazem que a fusão de multi sensores é uma tecnologia que realiza a combinação sinérgica de dados sensoriais de múltiplos sensores a fim de atingir inferências que não são possíveis com os sensores operando separadamente.

A ideia de unir sensores não é recente na história da humanidade, um exemplo muito bom para mostrar esse fato é que, de acordo com Hall e Llinas (1997), humanos e animais desenvolveram a capacidade de utilizar múltiplos sentidos para melhorar suas habilidades de sobrevivência, como no caso de um animal que não consegue ver ao redor de cantos ou por meio da vegetação, então o sentido de audição pode prover bons avisos de perigos. Assim, a fusão de sensores é naturalmente realizada por animais e humanos para uma melhor abordagem do ambiente ao redor e para identificação de ameaças.

De acordo com MÁRTON e GYÖRGY (2013), a fusão de sensores é um método efetivo para solucionar o problema de localização precisa de robôs móveis. Nessa técnica, mais de um sensor é utilizado para obter a posição do robô e para uma combinação efetiva de diferentes medições a fim de gerar os estados estimados do sistema. Assim, a fusão de sensores permite a mitigação das limitações de diferentes sensores, obtendo uma posição mais precisa do robô.

A seguir, nessa seção serão comentadas as vantagens e desvantagens de utilizar fusão de sensores em sistemas inteligentes, além de apresentar três tipos de classificação dos diferentes métodos e técnicas de fundir dados de múltiplos sensores em um sistema para melhor entendimento da técnica utilizada neste projeto.

### 3.4.1 Vantagens e Desvantagens

Fung, Chen e Chen (2017) trazem que a maioria dos sensores não geram diretamente um sinal de um fenômeno externo, mas sim através de diversas etapas de conversão. Por conta disso, o dado sensorial lido pelo usuário pode desviar da entrada real. O autor também coloca que existem algumas características de sensores que são inevitáveis, como velocidade e frequência de resposta, atraso e tempo de acomodação, e que acabam levando a diversas complicações, que são enfrentadas pela fusão de sensores. Além disso, existem outras características estáticas, como acurácia, precisão, resolução e sensibilidade, que podem ser facilmente geridas antes do processo de fusão.

Fung, Chen e Chen (2017) trazem também que a maior parte dos sensores não são ideais e possuem desvios que podem vir junto da informação necessária, alguns deles podem ser considerados de uma fonte aleatória de ruído, que precisam de um processamento para redução, já outros são considerados sistemáticos correlacionados com o tempo, estes também podem ser melhorados se o erro é conhecido.

Como comentado anteriormente, o principal propósito de sensores externos é prover ao sistema informação útil no que diz respeito a informações de interesse do ambiente. A fusão de dados de diferentes sensores traz diversas vantagens relacionadas a obtenção de informações mais precisas, que no caso são impossíveis de perceberem somente com os dados individuais. Segundo Alatise e Hancke (2020), os seguintes itens são as principais vantagens da fusão de dados de sensores.

- a) **Redução da incerteza:** os dados providenciados por sensores estão, por vezes, sujeitos a um nível de incerteza e discrepância. Assim, a fusão de dados de diferentes sensores reduz a incerteza ao combinar dados de inúmeras fontes. É, assim, imperativo compensar usando outros sensores ao fundir seus dados utilizando algoritmos de fusão.
- b) **Aumento na acurácia e confiabilidade:** integração de múltiplos sensores vai permitir que o sistema providencie informação inerente mesmo em caso de falha parcial em algum de seus módulos sensoriais.
- c) **Cobertura temporal e espacial estendida:** a área coberta por um sensor pode não ser coberta por outro sensor, portanto a medição de um é dependente do outro e ambos se complementam. Um exemplo em que ocorre isso é um sensor inercial, como acelerômetro e giroscópio, e visão, nesse caso a cobertura da câmera como

sensor de visão não pode ser comparada com o uso do acelerômetro, que apenas pega medidas sobre a rota de navegação.

- d) **Resolução aprimorada:** o valor da resolução resultante de múltiplas medições independentes fundidas é melhor que a medição singular de um sensor.
- e) **Complexidade reduzida do sistema:** um sistema em que os dados do sensor são pré-processados por algoritmos de fusão, a entrada para a aplicação de controle pode ser padronizada de forma autônoma dos tipos de sensores empregados, assim simplificando a implementação e providenciando a opção de modificações no sistema de sensor relativo ao número e tipo dos sensores sem alterações do software aplicado.

Embora seja provado a qualidade da fusão de sensores, de acordo com Fung, Chen e Chen (2017), existem alguns problemas associados com a criação de uma metodologia geral para fusão de diferentes sensores e eles se concentram em torno dos métodos utilizados para modelagem do erro ou incertezas no processo de integração dos dados, na informação sensorial e na operação do sistema em geral incluindo os sensores. Sendo assim, os seguintes itens são colocados pelo autor como potenciais problemas.

- a) **Registro dos dados:** sensores individuais possuem seus próprios frames de referência do qual os dados são calculados. Para que a fusão ocorra, os conjuntos de dados diferentes devem ser convertidos para um frame de referência comum, e assim alinhados juntos. Erro de calibração de sensores individuais deve ser abordado durante este estágio. Este problema acaba sendo crítico na determinação se a fusão funcionará ou não.
- b) **Incerteza nos dados sensoriais:** Diversos formatos de dados podem criar ruídos e ambiguidade no processo de fusão. Dados competitivos ou conflitivos podem ser resultados desses erros. A redundância dos dados de diversos sensores precisa estar engajada em reduzir a incerteza e aprender a rejeitar valores discrepantes se dados conflitivos são encontrados.
- c) **Dados incompletos, inconsistentes e falsos:** dados são considerados incompletos se os dados observados permanecem os mesmos independente do número de interpretações. Sensores inconsistentes são definidos como dois ou mais conjuntos de dados completos mas que possuem diferentes interpretações.
- d) **Associação de dados/Correspondência:** um aspecto da fusão de sensores é estabelecer se duas faixas de cada sensor representam o mesmo objeto, sendo

isto necessário para saber como características de dados combinam de diferentes sensores, além de saber se podem ser discrepantes.

- e) **Granularidade:** o nível de detalhes de diferentes sensores são dificilmente similares. Os dados podem ser esparsos ou densos, relativos a outros sensores. O nível dos dados pode ser diferente e isso deve ser abordado no processo de fusão.
- f) **Escalas de tempo:** sensores podem medir o mesmo ambiente em taxas diferentes. O tempo de chegada ao nó de fusão pode não coincidir por conta de atrasos de propagação no sistema. Em casos em que o algoritmo de fusão necessita do histórico dos dados, o quanto rápido o sensor consegue prover o dado é diretamente relacionado à validade dos resultados.

### **3.4.2 Classificação de técnicas**

Após o entendimento do que é a fusão de sensores, como ela pode ajudar diferentes sistemas e alguns pontos dela que merecem certa atenção para evitar problemas, é necessário diferenciar as diversas técnicas que realizam essa função de unir dados de sensores. De acordo com Castanedo et al. (2013), esse tema é uma área multidisciplinar que envolve diferentes campos do conhecimento, portanto é difícil estabelecer uma classificação clara e estrita das diferentes técnicas. Por isso, foram escolhidas 3 maneiras para classificação dos diferentes métodos de fusão sensorial.

#### ***3.4.2.1 Classificação baseada na relação entre as fontes de dados***

De acordo com Castanedo et al. (2013), a relação entre as fontes de dados é uma maneira de dividir as diferentes técnicas de fusão de sensores, separando nas seguintes três categorias.

- a) **Complementar:** é o caso de quando os sensores não dependem diretamente entre si, mas podem ser combinados de uma maneira que entreguem uma visão mais completa do fenômeno sendo observado. Ou seja, a informação providenciada pelas diferentes fontes representam diferentes partes do cenário. Um exemplo que pode ser colocado são câmeras em uma sala sendo que cada uma acaba observando partes disjuntas.
- b) **Competitiva:** também chamada de redundante, é o caso em que cada sensor entrega medidas independentes de uma mesma propriedade e, assim, as informações podem ser utilizadas a fim de obter uma informação global mais confiável. Visser e Groen

(1999) ainda separam essa categoria em dois - a fusão de dados de diferentes sensores ou a fusão de medições de um mesmo sensor tomadas em diferentes instantes. Um exemplo é o caso de dados vindo de áreas sobrepostas em redes de sensores visuais.

- c) **Cooperativa:** é quando as informações fornecidas por dois sensores independentes são utilizadas para conseguir alguma informação que não estaria disponível com os sensores funcionando sozinhos. De acordo com Brooks e Iyengar (1998), é a fusão mais difícil de projetar, já que o dado resultante está suscetível a problemas de todos os sensores sendo fundidos, o que geralmente diminui a acurácia e confiabilidade em relação às outras categorias. Um exemplo é uma fusão de dados multi-modal entre áudio e vídeo para gerar uma informação mais complexa.

De acordo com essa classificação, esse projeto transitou entre os tipos complementar e competitiva, já que depende da combinação de sensores a ser empregada. Por exemplo, o encoder e o giroscópio podem prover observações sobre a velocidade angular, o que é uma fusão competitiva, mas há combinações que estão no caso de fusão complementar por apresentarem dados diferentes de uma mesma cena.

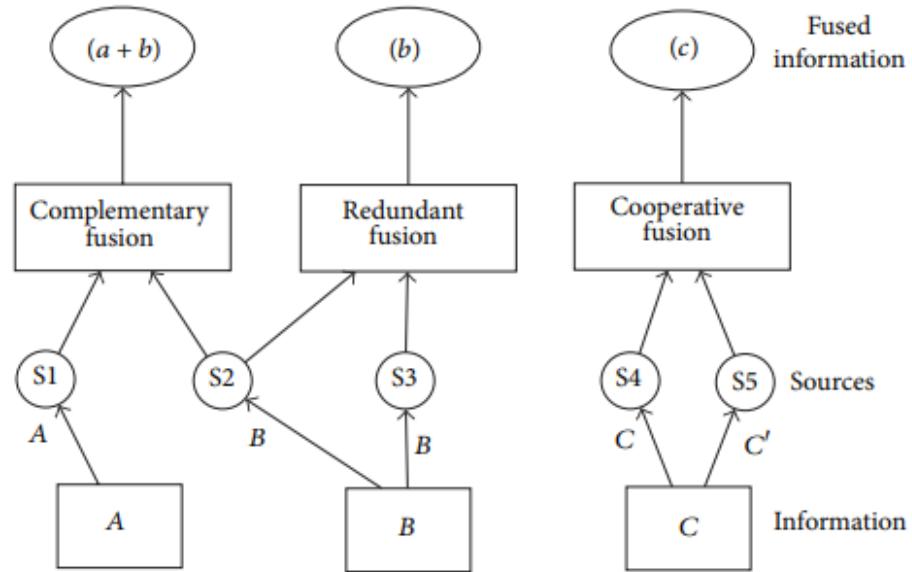
A Figura 6 representa claramente a diferença entre as três categorias da classificação proposta por Castanedo et al. (2013).

#### ***3.4.2.2 Classificação em três níveis***

A fusão de dados normalmente aborda três níveis de abstração: medidas, características e decisões. Essa maneira de classificar os diferentes métodos de fusão de sensores se baseia nessa ideia, dividindo-os em baixo, intermediário e alto nível, como é descrito a seguir (CASTANEDO et al., 2013).

- a) **Nível baixo:** também chamada de fusão de dados crus, essa categoria combina diferentes fontes de dados crus para produzir novos dados que esperase que sejam mais informativos do que os de entrada.
- b) **Nível intermediário:** também chamada de fusão a nível de características, essa categoria combina diversos aspectos, como bordas, linhas, texturas ou posições em um mapa de características que pode ser utilizado para segmentação ou detecção.

Figura 6 – Diagrama representando a diferença entre as fusões complementar, competitiva e cooperativa.



Fonte: Retirado de Castanedo et al. (2013)

- c) **Nível alto:** também chamada de fusão de decisões, essa categoria combina decisões de diversos especialistas para obter uma decisão ainda mais precisa. Normalmente métodos bayesianos são empregados neste nível.
- d) **Nível múltiplo:** esse nível aborda dados providenciados de diferentes níveis de abstração. Um exemplo é a união de uma medição com uma característica para obter uma decisão.

Segundo essa classificação, esse projeto utiliza uma fusão em baixo nível, visto que utiliza dados crus de sensores a fim de predizer a localização de um robô móvel, que é um dado mais informativo do que os dados de entrada.

#### 3.4.2.3 Classificação baseada na entrada e saída do sistema

Esse sistema de classificação proposto por Dasarathy (1997) (por isso também chamado de modelo de Dasarathy) refinou o modelo de classificação em três níveis, dividindo as diferentes técnicas de fusão de sensores em 5 categorias baseado no nível de abstração dos dados de entrada e saída do sistema. As características das 5 categorias são resumidas de acordo com a natureza da entrada e os resultados de saída do processo de fusão (LI et al., 2021; VAKIL et al., 2021).

- a) **Entrada de dados-Saída de dados (DAI-DAO):** é o tipo mais básico de fusão de dados, nele há o processo de entradas e saídas de dados crus, os resultados são tipicamente mais confiáveis e acurados. Nesse caso a fusão é conduzida imediatamente depois da coleta de dados dos sensores.
- b) **Entrada de dados-Saída de características (DAI-FEO):** nesse nível o processo de fusão emprega dados crus das fontes para extrair características ou aspectos que descrevem a entidade do ambiente.
- c) **Entrada de características-Saída de características (FEI-FEO):** nesse nível tanto a saída quanto a entrada dos dados do processo de fusão são características. Assim, o processo aborda um conjunto de aspectos a fim de melhorar, refinar ou obter novas características.
- d) **Entrada de características-Saída de decisões (FEI-DEO):** este nível recebe um conjunto de características e, a partir delas, fornece um conjunto de decisões como saída do sistema. A maior parte dos sistemas que realizam uma decisão baseada no recebimento de dados de sensores entram nessa categoria.
- e) **Entrada de decisões-Saída de decisões (DEI-DEO):** esse tipo de classificação é também conhecida como fusão de decisão, já que funde decisões de entrada para obtenção de melhores ou novas decisões.

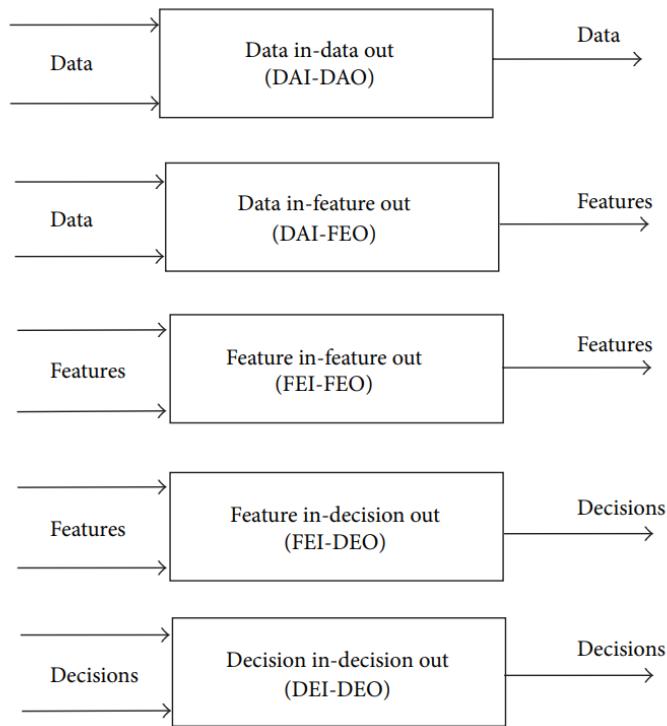
A partir dessa classificação, pode-se dizer que a fusão de sensores desse projeto funcionou com entrada de dados e saída de dados, já que foram utilizados dados de posição, aceleração e velocidade para conseguir melhores dados de posição e orientação.

A Figura 7 representa claramente a diferença entre as cinco categorias de classificação proposta por Dasarathy (1997). Já a Figura 8 relaciona e mostra as diferenças entre os modelos de classificação em três níveis e Dasarathy.

### 3.5 FILTRO DE KALMAN

Nesta seção serão abordados os conceitos teóricos necessários para entendimento do tão divulgado e utilizado filtro de Kalman. Nela, serão apresentadas suas equações, as premissas para desenvolvimento das equações, as etapas do algoritmo, assim como a apresentação do filtro de Kalman estendido, utilizado em situações cujo sistema e/ou medição são não-lineares.

Figura 7 – Diagrama representando a diferença entre as fusões baseadas no nível de abstração dos dados.



Fonte: Retirado de Castanedo et al. (2013)

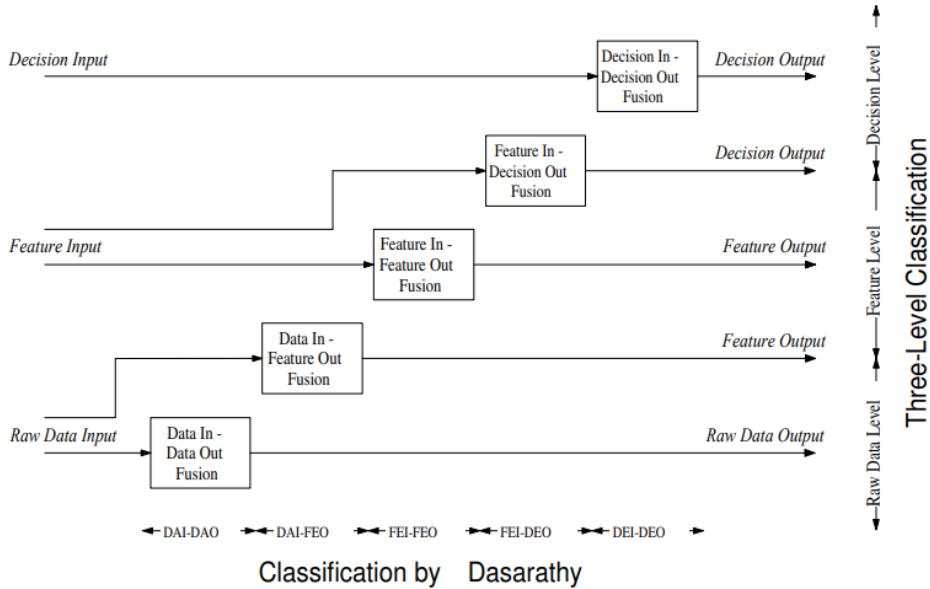
No caso desse trabalho, o filtro de Kalman é o algoritmo utilizado para realizar a fusão de diferentes sensores a fim de estimar a localização de robôs móveis, por isso a importância da descrição detalhada do seu funcionamento ao longo dessa seção.

### 3.5.1 Utilização

O filtro de Kalman foi inventado durante a década de 50 por Rudolph Emil Kalman como uma técnica para filtragem e predição em sistemas lineares. Desde então, por conta dos avanços na área de computação digital, o KF é objeto de extenso estudo e aplicações, particularmente na área de navegação autônoma ou assistida.

O filtro de Kalman é um algoritmo que já foi utilizado em uma vasta gama de aplicações, principalmente na área de controle e na predição de sistemas dinâmicos, sendo a base para o desenvolvimento da teoria do controle moderno e processamento de sinais em tempo real. Nos dias de hoje, segundo Khodarahmi e Maihami (2023), o KF evoluiu de um simples estimador

Figura 8 – Diagrama relacionando as classificações três níveis e Dasarathy.



Fonte: Retirado de Elmenreich (2002)

de estados ótimo e possui aplicações na automação, posicionamento, rastreamento de alvo, processamento de sinais, imagens digitais, sinais de voz e previsão de terremotos.

Focando mais no campo da robótica, o filtro de Kalman é aplicado no rastreamento de trajetória, estimativa de posição para robôs manipuladores, SLAM (do inglês, *Simultaneous Localization and Mapping*) e detecção de objetos (URREA; AGRAMONTE, 2021). Além de que sua flexibilidade permitiu a integração da informação de diferentes tipos de sensores e técnicas, tornando possível responder as questões fundamentais da navegação de robôs: Onde estou? Para onde estou indo? E como eu chego no meu destino?

Em suma, o filtro de Kalman é um conjunto de equações matemáticas que serve para estimar o estado de um sistema dinâmico linear com ruídos de tal maneira que a média do erro quadrático diminui de forma eficiente computacionalmente por ser um algoritmo recursivo. Ou seja, o KF precisa de pouca memória já que precisam de memória apenas para salvar informação de estados passados, sendo adequado para problemas de tempo real e sistemas embarcados (KHODARAHMI; MAIHAMI, 2023).

Quando fala-se sobre o estado de um sistema, coloca-se um vetor  $x$  que consiste de  $n$  variáveis que descrevem importantes propriedades de um sistema. Um exemplo de estado é a pose de um robô, que consiste das coordenadas  $x$  e  $y$  e a orientação  $\theta$  de um robô.

Como colocado anteriormente, robôs normalmente utilizam uma grande quantidade de sensores, cada um deles provendo a posição do robô, mas também cada um sendo sujeito a erros ou falhas no funcionamento. Portanto, a obtenção da localização ótima de um robô móvel deve levar em conta a informação gerada por todos sensores. Segundo Siegwart, Nourbakhsh e Scaramuzza (2011), o filtro de Kalman é uma técnica poderosa para atingir essa fusão de sensores por ser eficiente ao representar a função de densidade probabilística da crença do robô e até das leituras individuais dos sensores, resultando num algoritmo de processamento de dados recursivo ótimo.

Entretanto, segundo Negenborn (2003), o fato de que as variáveis de um estado podem conter ruídos e não serem diretamente observáveis dificultam a estimativa do estado. O KF possui acesso às medições do sistema para poder realizar a estimativa do estado, estas medições estão linearmente relacionadas ao estado e estão corrompidas por ruídos. Caso as fontes desses ruídos possuírem uma distribuição gaussiana, a estimativa do KF é estatisticamente ótima para qualquer medida razoável de otimização.

Também segundo Negenborn (2003), o KF processa todas medidas disponíveis de sensores para estimar o estado, tanto as medidas precisas quanto as imprecisas. Ele utiliza conhecimento do sistema e dinâmica dos sensores, descrição probabilística do próprio sistema e dos ruídos das medidas, e qualquer dado disponível sobre os valores iniciais do estado.

### **3.5.2 Premissas**

A utilização do filtro de Kalman para predizer e corrigir a crença do estado presume a necessidade de um modelo tanto do sistema quanto das medições. O KF assume uma descrição de sistema dinâmico linear do sistema que está estimando o estado. O sistema dinâmico pode ser corrompido por fontes de ruídos, os quais o KF assume que podem ser modelados por distribuições independentes, brancas, média zero e gaussianas (URREA; AGRAMONTE, 2021).

#### **3.5.2.1 Sistema dinâmico linear**

Falando sobre o modelo do sistema, ele descreve como o verdadeiro estado do sistema evolui ao longo do tempo, utilizado pelo filtro para realizar previsões sobre o estado. Basicamente, o KF assume que o estado do sistema evolui de acordo com a Equação (8), onde o verdadeiro estado  $x_k$  do sistema no tempo  $k$  depende do estado um passo antes  $x_{k-1}$  e algum ruído, a matriz  $A$  tem tamanho  $n \times n$  e relaciona os estados passado e atual, enquanto o vetor  $w_k - 1$  modela

o ruído no sistema, adicionando os efeitos de influências não modeladas no estado (URREA; AGRAMONTE, 2021).

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad (8)$$

O modelo de medição descreve como medições se relacionam com os estados. O filtro de Kalman necessita do modelo das medições a fim de corrigir a predição do estado quando uma medição está disponível. Basicamente, o KF assume que as medições podem ser modeladas por um equação linear que relaciona o estado do sistema para uma medição, como a Equação (9), em que  $z_k$  depende linearmente do estado do sistema  $x_k$ , já a matriz H possui tamanho  $m \times n$  e relaciona a medição com o estado, enquanto  $v_k$  é o ruído nas medições (KHODARAHMI; MAIHAMI, 2023).

$$z_k = Hx_k + v_k \quad (9)$$

Ambas Equações (8) e (9) mostram que o estado  $x_k$  no tempo  $k$  não depende de todos os outros estados e medições dado  $x_{k-1}$  e que a medição  $z_k$  não depende de qualquer estado ou medida, o que torna o sistema um processo Markoviano.

### 3.5.2.2 Características do ruído

Uma característica necessária do ruído para o filtro de Kalman é a independência, que torna o cálculo envolvido na estimativa de estado mais fácil. De acordo com Negenborn (2003) em geral é justo assumir que os ruídos no sistema e medição são independentes.

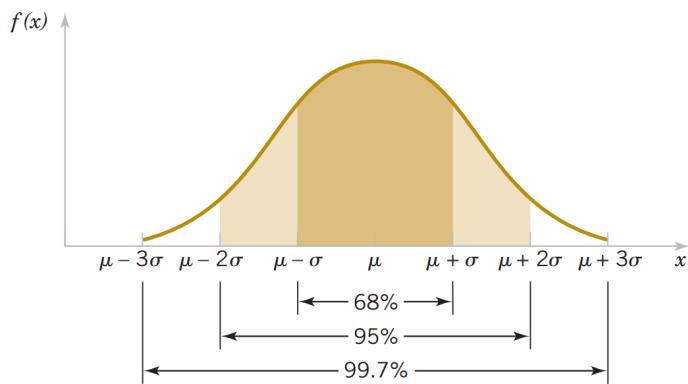
Outra característica que simplifica a matemática envolvida no filtro de Kalman é o ruído branco, este tem poder em todas frequências do espectro e é completamente não correlacionado com ele mesmo em qualquer momento exceto o presente. Ou seja, os erros não se correlacionam pelo tempo, saber a quantidade de ruído neste momento não ajuda em predizer qual será a quantidade de ruído em outro momento.

Uma terceira característica que é assumida é que o ruído possui média zero, o que implica que o erro no sistema e medição é aleatório. Um ruído aleatório significa que ele não é sistemático, ou seja, ele não possui um *bias* constante, algumas horas ele é positivo, outras negativo, mas sempre média zero.

A última característica importante que é assumida pelo filtro de Kalman é que o ruído é gaussiano, que é uma característica que lida com amplitude do ruído, colocando que a quantidade

de ruído envolvida pode ser modelada por uma curva conforme a Figura 9, em que o centro do gráfico representa a média  $\mu$  dos valores, já a dispersão (ou largura) do gráfico é representada pelo desvio padrão  $\sigma$  (ou pela variância, que é o desvio padrão elevado ao quadrado). Esta premissa é justificada ao assumir que os ruídos do sistema e medição são causados por diversas fontes pequenas de ruídos que, independente de suas distribuições, a soma delas será distribuída conforme uma gaussiana.

Figura 9 – Exemplo de distribuição gaussiana.



Fonte: Retirado de Zibetti (2022)

Com as premissas da média zero e a distribuição gaussiana, os ruídos podem ser descritos de acordo com  $N(\mu, \Sigma)$ , que denota uma função gaussiana de média  $\mu$  e covariância  $\Sigma$ .

### 3.5.2.3 Processo a ser estimado

O filtro de Kalman aborda o problema geral de tentar estimar o estado  $x \in \mathbb{R}^n$  de um processo controlado em tempo discreto que é governado pela equação diferencial estocástica linear descrita pela Equação (10) com medição  $z \in \mathbb{R}^m$ , que é representada pela Equação (11). No caso, as variáveis aleatória  $w_k$  e  $v_k$  representam os ruídos do processo e das medições, respectivamente.

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad (10)$$

$$z_k = Hx_k + v_k \quad (11)$$

Assume-se que as variáveis  $w_k$  e  $v_k$  são independentes uma das outras, brancas, e com uma distribuição probabilística normal, segundo as probabilidades definidas na Equação (12).

$$\begin{aligned} p(w) &\sim N(0, Q) \\ p(v) &\sim N(0, R) \end{aligned} \tag{12}$$

Segundo Welch, Bishop et al. (1995), a matriz de covariância do ruído do processo Q e a matriz de covariância do ruído das medições R podem variar a cada passo de tempo ou a cada medição, embora nesse caso seja considerado constante.

A matriz A da Equação (10) possui tamanho  $n \times n$  e relaciona o estado no passo de tempo anterior  $k - 1$  com o estado no passo de tempo atual  $k$  na ausência de uma função ou ruído de processo. Já a matriz B possui tamanho  $n \times l$  e relaciona a entrada de controle  $u \in \mathbb{R}^l$  ao estado  $x$ . A matriz H possui tamanho  $m \times n$  na Equação (11) e relaciona o estado com a medição  $z_k$ .

### 3.5.3 Equações

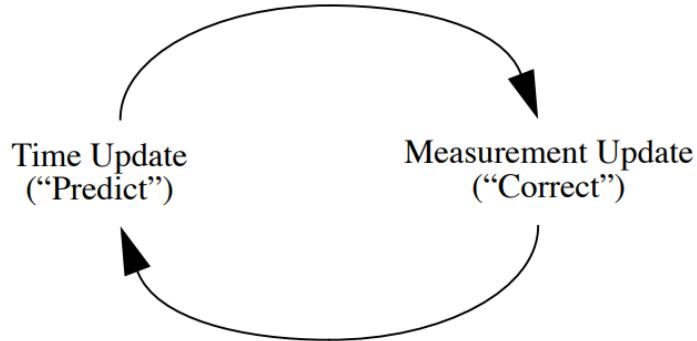
De acordo com Khodarahmi e Maihami (2023), o filtro de Kalman estima um processo utilizando uma forma de controle por meio de feedback, nele o filtro estima o estado do processo em um dado instante e obtém feedbacks na forma de medições, no caso ruidosas. Como tal, as equações do filtro de Kalman podem ser divididas em dois grupos: as equações de atualização de tempo e as equações de atualização de medições. O primeiro grupo é responsável por projetar a frente no tempo as estimativas do estado atual e a covariância do erro para obter a estimativa a priori do próximo período de tempo. Já o segundo grupo é responsável pelo feedback, isto é, por incorporar uma nova medição na estimativa a priori a fim de obter uma melhor estimativa a posteriori.

As equações de atualização no tempo podem ser chamadas como equações de predição, enquanto as equações de atualização de medição podem ser chamadas de equações de correção. Basicamente o algoritmo de estimativa final se assemelha com um algoritmo predição-correção para solução de problemas numéricos. A Figura 10 mostra o ciclo do filtro de Kalman, em que a predição projeta a estimativa do estado atual a frente no tempo, enquanto a correção ajusta a estimativa projetada por uma medição real naquele instante.

#### 3.5.3.1 Predição

A cada instante de tempo o sistema pode estar em um estado diferente. Portanto, o KF calcula uma nova crença anterior a cada passo de tempo. As equações de predição (também chamada de atualização por tempo ou propagação) predizem o novo estado do sistema projetando

Figura 10 – Ciclo do filtro de Kalman discreto.



Fonte: Retirado de Welch, Bishop et al. (1995)

à frente a crença mais recente, ou seja, calculando a crença  $bel(x_k)$  a partir da crença do estado anterior  $bel(x_{k-1})$ .

No caso, de acordo com Thrun (2002),  $bel(x_k) = N(\hat{x}_k^-, P_k^-)$ , em que a média  $\hat{x}_k^-$  e a covariância  $P_k^-$  são definidos segundo a Equação (13). É importante destacar que a notação  $\hat{\cdot}$  representa uma estimativa do estado, enquanto a notação  $^-$  representa que a variável foi predita, ou seja, antes de incorporar a medição.

$$\begin{aligned}\hat{x}_k^- &= A\hat{x}_{k-1} + Bu_k \\ P_k^- &= AP_{k-1}A^T + Q_k\end{aligned}\tag{13}$$

O KF calcula a estimativa de estado  $\hat{x}_k^-$  baseado tanto na última estimativa de estado  $\hat{x}_{k-1}$  quanto no modelo disponível do sistema. A melhor hipótese que o KF pode fazer sobre o estado do sistema depois dele progredir um passo a frente no tempo é a melhor hipótese propagada pelo modelo que o KF possui do sistema.

Além disso, o filtro de Kalman também reconhece que a evolução do sistema está sujeita a ruídos e, assim, possui uma incerteza aumentada  $P_k^-$  na estimativa do estado. O primeiro termo da covariância do erro  $AP_{k-1}A^T$  propaga a incerteza da última estimativa à frente para a estimativa atual do estado. Já o segundo termo  $Q_k$  é o ruído do sistema que corrompe o estado do sistema a cada passo de tempo.

### 3.5.3.2 Correção

As equações de correção (ou atualização da medição) lidam com as medições dos sensores. Elas são utilizadas apenas quando há a atualização da medição dos sensores. As medições providenciam informação direta sobre o estado atual do sistema. As equações desta etapa corrigem a previsão da crença mais recente ao incorporar a informação recebida das medições. Segundo Thrun (2002), as equações dessa fase calculam a crença posterior  $bel(x_k) = N(\hat{x}_k, P_k)$ , em que  $\hat{x}_k$  e  $P_k$  são definidos segundo a Equação (14).

$$\begin{aligned}\hat{x}_k &= \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \\ P_k &= (I - K_k H)P_k^- \\ K_k &= P_k^- H^T (H P_k^- H^T + R_k)^{-1}\end{aligned}\tag{14}$$

A nova crença posterior  $\hat{x}_k$  é utilizada no próximo passo de tempo para calcular a previsão de uma nova crença. A natureza recursiva do filtro de Kalman permite implementações práticas, já que nem todos os dados são necessários para estimar os estados.

O filtro de Kalman calcula a estimativa do estado posterior combinando a previsão da estimativa de estado com o ganho de Kalman  $K_k$  vezes a diferença entre a medição  $z_k$  e a previsão de medição  $H\hat{x}_k^-$ , chamada de inovação.

O termo  $H\hat{x}_k^-$  na Equação (14) é chamado de previsão de medição. Dadas a estimativa do estado anterior  $x_k^-$  e a matriz de medições  $H$  do modelo de medição na Equação (11), o filtro de Kalman prediz qual medição irá receber. Assim, denota-se a previsão de medição segundo a Equação (15).

$$\hat{z}_k = H\hat{x}_k^- + \hat{v}_k\tag{15}$$

No caso, o ruído de medição  $\hat{v}_k$  é zero e a previsão de medição é uma variável aleatória que segue uma distribuição gaussiana, podendo notar isso ao analisar que ela depende linearmente da estimativa anterior do estado  $\hat{x}_k^-$  e do ruído de medição, sendo que ambos são variáveis aleatórias gaussianas. Logo, facilmente deriva-se que a previsão de medida  $\hat{z}_k$  segue a distribuição descrita na Equação (16).

$$\hat{z}_k = N_z(H\hat{x}_k^-, H P_k^- H^T + R_k)\tag{16}$$

A diferença entre a medição  $z_k$  e a medição prevista  $x_k^-$  é chamada de inovação da medição ou  $\tilde{z}_k$  residual. A inovação diz quanto uma medida prevista difere de uma medição real,

sendo definida segundo a Equação (17). Caso a inovação seja igual a zero, a medida prevista reflete exatamente a medição real, o que implica que o estado estimado com o qual a predição da medição foi realizada estava muito perto do verdadeiro estado que a medição foi feita. Entretanto, se existir uma diferença entre as medições prevista e observada, a estimativa do estado anterior precisa ser atualizada com um certo valor.

$$\tilde{z}_k = z_k - \hat{z}_k \quad (17)$$

O fator  $K_k$  na Equação (14) é chamado de ganho de Kalman (KG, do inglês *Kalman Gain*), que é o fator que determina até que ponto a inovação deve ser levada em conta na estimativa de estado posterior. Isso é determinado ao olhar a incerteza relativa entre a estimativa de estado anterior e a inovação da medição, como descrito na Equação (14).

A fim de comparar a incerteza da estimativa do estado anterior no espaço de estados com a incerteza da inovação no espaço de medição, o KF converte a incerteza no espaço de medição para o espaço de estados por meio da matriz  $H^T$ .

Em resumo, o ciclo do filtro de Kalman pode ser entendido conforme o Algoritmo 1, em que há o detalhamento das Equações de cada um dos passos do KF, a predição e correção. Já a Figura 11 representa um exemplo em uma dimensão de como o filtro de Kalman realiza a predição e correção em termos de média e covariância.

#### Algoritmo 1 – Filtro de Kalman linear

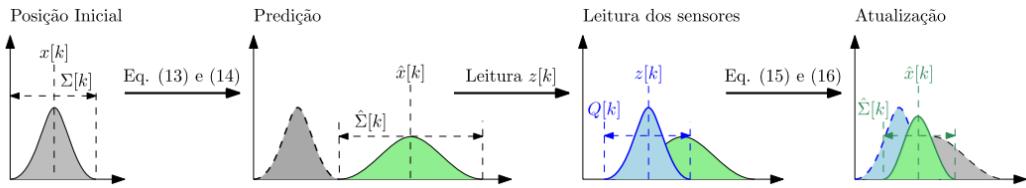
```

1 Entrada: Estado anterior  $x_{k-1}$ ; Covariância anterior  $P_{k-1}$ ; Entrada de controle  $u_k$ ;
   Entrada de medição  $z_k$ 
2 Saída: Estado atual  $\hat{x}_k$ ; Covariância atual  $P_k$ 
3 início
4    $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$ 
5    $P_k^- = AP_{k-1}A^T + Q_k$ 
6   se medição disponível então
7      $K_k = P_k^- H^T (H P_k^- H^T + R_k)^{-1}$ 
8      $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$ 
9      $P_k = (I - K_k H)P_k^-$ 
10  fim
11 fim
12 retorna  $\hat{x}_k$ ,  $P_k$ 
```

### 3.6 FILTRO DE KALMAN ESTENDIDO

Para sua descrição, o filtro de Kalman possui algumas premissas, tal como a de que as observações são funções lineares do estados e que o próximo estado é uma função linear do

Figura 11 – Exemplo de predição e atualização das covariâncias do filtro de Kalman.



Fonte: Retirado de Costa e Tonidandel (2023)

estado anterior, o que é crucial para a correção do filtro (THRUN, 2002). Não só isso, para o desenvolvimento do KF, observa-se que qualquer transformação linear de uma variável aleatória gaussiana resulta em outra variável aleatória gaussiana.

Embora o filtro de Kalman tenha provado sua eficiência e qualidade ao longo dos anos com sua vasta utilização em diversas áreas, infelizmente sistemas mais complicados podem ser não-lineares (KHODARAHMI; MAIHAMI, 2023). Por exemplo, um robô que se move com velocidade de translação e rotação constantes tipicamente realizam uma trajetória circular, que não pode ser descrita por uma transição de estado linear (THRUN, 2002).

Logo, a fim de resolver o problema da linearidade para o filtro de Kalman, foi desenvolvida uma versão dele que leva em conta a não-linearidade dos sistemas, medições e ruídos, que é o filtro de Kalman estendido (EKF, do inglês).

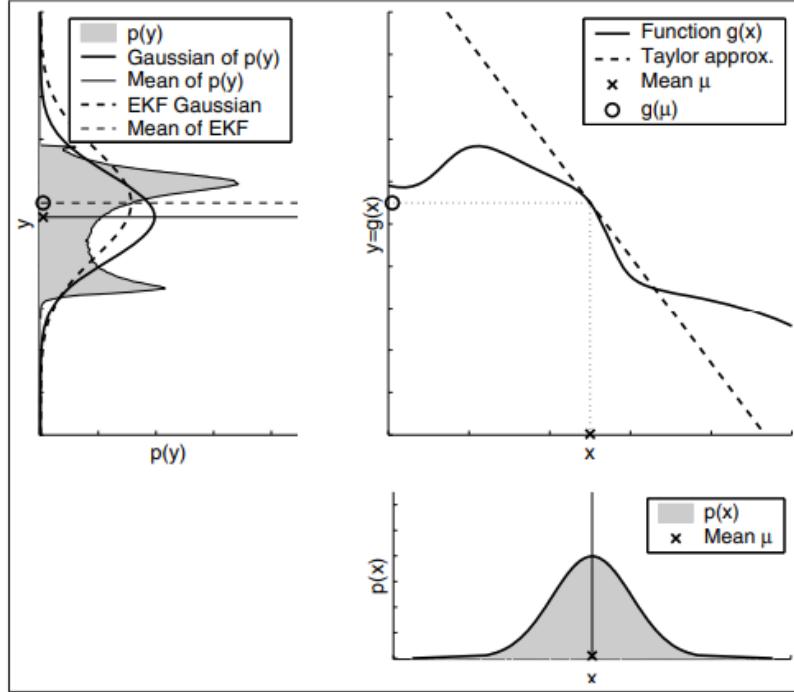
O EKF segue a mesma ideia do filtro de Kalman linear, isto é, com a separação nas etapas de predição, que projeta o sistema a frente para obter uma estimativa no próximo período de tempo, e correção, que incorpora uma nova medição na estimativa da predição a fim de obter uma melhor estimativa. A diferença entre ambos recai na particularidade de que o EKF utiliza séries de Taylor para linearizar o sistema não-linear.

De acordo com Thrun (2002), a ideia da linearização é aproximar uma função não-linear  $g$  por uma função linear que é tangente a  $g$  na média da gaussiana. Assim, projetar a gaussiana por meio dessa aproximação linear resulta em uma densidade gaussiana, como é demonstrado na Figura 12. O autor coloca que a principal vantagem da linearização recai na sua eficiência.

As previsões lineares no filtro de Kalman são substituídas pelas generalizações não-lineares no filtro de Kalman estendido. Além de que o EKF utiliza de Jacobianas  $G_k$  e  $C_k$  ao invés das matrizes lineares do sistema  $A_k$ ,  $B_k$  e  $H_k$  no KF.

Assim, a etapa de predição no filtro de Kalman estendido é descrita conforme a Equação (18). É possível notar que a estimativa do estado é dada por uma função não-linear  $f$  que depende

Figura 12 – Exemplo de predição e atualização das covariâncias do filtro de Kalman.



Fonte: Retirado de Thrun (2002)

do estado anterior  $x_{k-1}$  e da entrada de controle naquele instante  $u_k$ . Além disso, como dito anteriormente,  $G_k$  é a matriz Jacobiana com relação ao estado. No caso, uma matriz Jacobiana é formada pelas derivadas parciais de primeira ordem de uma função. A variável  $P$  representa a covariância dos estados e  $Q$  os ruídos do sistema.

$$\begin{aligned}\hat{x}_k^- &= f(x_{k-1}, u_k) \\ P_k^- &= G_k P_{k-1} G_k^T + Q_k \\ G_k &= \frac{\partial f(x_{k-1}, u_k)}{\partial x_{k-1}}\end{aligned}\tag{18}$$

A etapa de correção no filtro de Kalman estendido é descrita conforme a Equação (19). No caso, a estimativa do sensor também é uma função não-linear. Assim,  $C_k$  representa a matriz Jacobiana do sensor com relação ao estado, já  $h(x_k^-)$  representa a linearização do sensor. A variável  $K$  representa o ganho de Kalman do sistema e  $R$  representa os ruídos das medições.

$$\begin{aligned}
z_k &= h(x_k) \\
\hat{x}_k &= \hat{x}_k^- + K_k(z_k - h(x_k^-)) \\
P_k &= (I - K_k C_k) P_k^- \\
K_k &= P_k^- C_k^T (C_k P_k^- C_k^T + R_k)^{-1} \\
C_k &= \frac{\partial h(x_k)}{\partial x_k}
\end{aligned} \tag{19}$$

As equações que representam o filtro de Kalman estendido podem ser entendidas em conjunto segundo o Algoritmo 2.

Algoritmo 2 – Filtro de Kalman estendido

```

1 Entrada: Estado anterior  $x_{k-1}$ ; Covariância anterior  $P_{k-1}$ ; Entrada de controle  $u_k$ ;
           Entrada de medição  $z_k$ 
2 Saída: Estado atual  $\hat{x}_k$ ; Covariância atual  $P_k$ 
3 início
4    $\hat{x}_k^- = f(x_{k-1}, u_k)$ 
5    $P_k^- = G_k P_{k-1} G_k^T + Q_k$ 
6   se medição disponível então
7      $K_k = P_k^- C_k^T (C_k P_k^- C_k^T + R_k)^{-1}$ 
8      $\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(x_k^-))$ 
9      $P_k = (I - K_k C_k) P_k^-$ 
10  fim
11 fim
12 retorna  $\hat{x}_k, P_k$ 

```

## 4 TRABALHOS RELACIONADOS

Diversos estudos foram realizados na área de localização de robôs móveis e estimativa de posição a fim de obter o conhecimento necessário para o desenvolvimento do projeto em questão. Entretanto, uma boa gama dos trabalhos encontrados utilizam robôs com uma dinâmica diferente de um robô omnidirecional da *Small Size League* da RoboCup, como o modelo de duas rodas ou o modelo *car-like*, ou também os estudos levam em conta diferentes sensores dos que são utilizados nesse projeto.

A seguir serão descritos os trabalhos relacionados nas áreas de fusão de sensores, estimativa de posição de robôs móveis e filtro de Kalman. Os termos utilizados na pesquisa dos trabalhos foram "*position estimation*", "*position estimation kalman filter*", "*sensor fusion for position estimation*" e "*position estimation sensor fusion*".

Em Eman e Ramdane (2020), um filtro de Kalman estendido é utilizado para resolver o problema de localização de um robô móvel num ambiente *indoor*. Os autores estudam a eficiência do filtro em três casos distintos na questão do ruído presente no sistema, que são: sem ruído, ruído Gaussiano, ruído não-Gaussiano.

Sobre o sistema, o modelo utilizado é de um robô de duas rodas, no qual as equações da cinemática do modelo estão descritas na Equação (20), que os autores definem  $(x, y)$  sendo a posição e  $\theta$  a orientação do robô,  $\mu$  a velocidade linear e  $\omega$  a velocidade angular.

$$\begin{cases} \dot{x} = \mu \cos \theta \\ \dot{y} = \mu \sin \theta \\ \dot{\theta} = \omega \end{cases} \quad (20)$$

A partir disso, utilizando aproximação de Taylor, a posição e a orientação do robô em qualquer momento futuro  $k + 1$  são descritas segundo a Equação (21), em que  $T_s$  é o período de amostragem.

$$\begin{cases} x(k + 1) = x(k) + \mu(k)T_s \cos(\theta(k)) \\ y(k + 1) = y(k) + \mu(k)T_s \sin(\theta(k)) \\ \theta(k + 1) = \theta(k) + \omega(k)T_s \end{cases} \quad (21)$$

Já sobre o EKF, o autor determina que o vetor de estados a serem estimados  $X$  e o vetor de controle  $U$  são o que está definido na Equação (22). Assim, as matrizes Jacobianias do sistema podem ser definidas conforme a Equação (23).

$$\begin{cases} X = [x \ y \ \theta]^T \\ U = [\mu \ \omega]^T \end{cases} \quad (22)$$

$$\begin{cases} A_k = \frac{\delta f(X_k, U_k)}{\delta X_k} = \begin{bmatrix} 1 & 0 & -\mu(k)T_s \sin(\theta(k)) \\ 0 & 1 & \mu(k)T_s \cos(\theta(k)) \\ 0 & 0 & 1 \end{bmatrix} \\ H_k = \frac{\delta f(X_k, U_k)}{\delta X_k} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{cases} \quad (23)$$

Para realização dos testes, os códigos foram desenvolvidos utilizando MATLAB num cenário 2D, com  $T_s$  sendo  $0.1s$ ,  $\mu$  sendo  $5m/s$  e  $\omega$  sendo  $1rad/s$ , além de que o vetor de estados inicial e a estimativa inicial são matrizes nulas. Como dito anteriormente, foram considerados 3 cenários de ruídos a fim de avaliar a atuação do EKF.

Os resultados mostraram que o EKF é uma boa ferramenta de estimativa para sinais ruidosos em todos os casos, embora os resultados tenham confirmado que o filtro funciona melhor no caso de ruídos Gaussianos comparado com ruídos não-Gaussianos. Essa conclusão é de suma importância para a realização desse trabalho, visto que os sinais dos sensores e do sistema deste projeto são assumidos como Gaussianos, além da importância do uso de diferentes tipos de ruídos para estudo.

Já em Korotaj, Novoselnik e Baotić (2021) é descrita a utilização de fusão de sensores para um sistema mecatrônico omnidirecional. As expressões são dadas para um filtro de Kalman linear discreto que junta dados de um magnetômetro e um giroscópio, e um filtro de Kalman estendido discreto que estima a posição e orientação da plataforma com dados de um acelerômetro também.

Sobre o sistema, a plataforma é composta de 2 níveis e possui 4 rodas acionadas para movimentação, cada uma acionada por um motor DC (do inglês, *Direct Current*). Os sensores presentes na plataforma são um acelerômetro de 3 eixos, um giroscópio de 3 eixos e um magnetômetro.

A estimativa de estado é separada em duas partes, a primeira é a estimativa da orientação do sistema a partir da fusão do giroscópio e do magnetômetro utilizando KF. Essa orientação estimada é utilizada como uma observação melhorada da orientação para estimativa tanto da

posição quanto da orientação na segunda parte por meio de um EKF discreto, já que o modelo cinemático da plataforma é não-linear, além de utilizar as velocidades de cada uma das rodas e as leituras do acelerômetro também.

Os resultados simulados da resposta de estimativa de estado mostraram satisfatórias acurácia e velocidade do procedimento de estimativa selecionado. Apesar das limitações do layout utilizado das rodas e imperfeições das ranhuras que conectam a roda o seu eixo do motor, os experimentos em tempo real confirmaram a eficiência da fusão de sensores.

O trabalho demonstrou uma maneira diferente da utilização do filtro de Kalman, já que ele é usado em um primeiro passo para estimar a orientação com os dados do magnetômetro para corrigir o erro da integração do giroscópio. Assim, a partir dessa estimativa, é possível utilizar o EKF para predizer tanto a posição quanto a orientação do veículo.

Em Ismail, Purwanto e Arifin (2022) é feito um estudo sobre a localização baseada em fusão de sensores de um robô de futebol da categoria *Middle Size League* (MSL) da *RoboCup*. A MSL é como se fosse uma categoria acima da SSL na questão de tamanho dos robôs, em que os robôs possuem carcaças maiores (de aproximadamente 1,30m X 30cm), além de que não há um sistema global de visão, ou seja, a visão deve estar embarcada em cada robô, assim cada robô deve estimar sua posição e orientação de forma automática e independente.

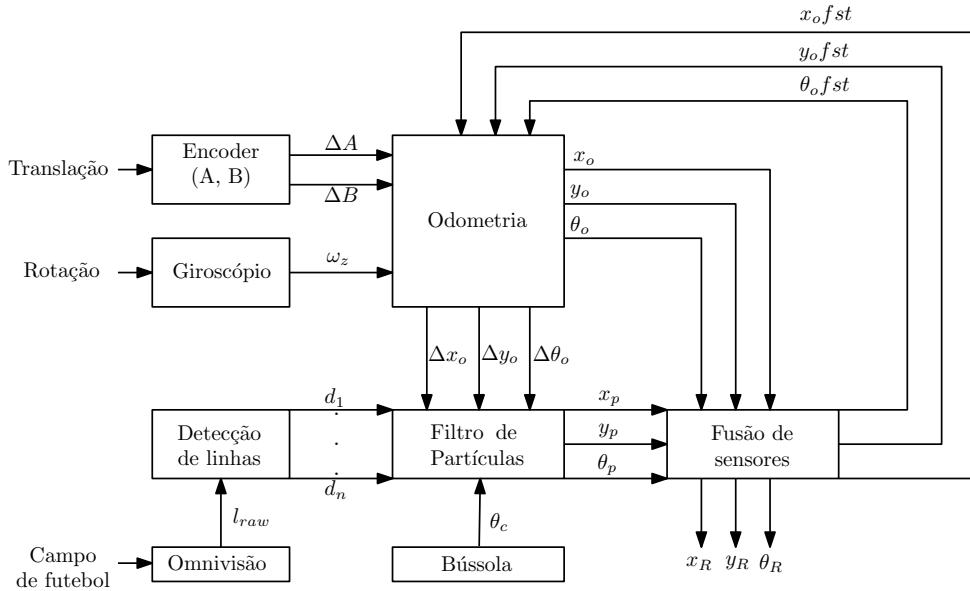
A maior parte dos robôs dessa categoria utilizam câmeras omnivisão, permitindo que os robôs tenham visão 360°. A partir dessas imagens é utilizado filtro de partículas (FP) para estimar a posição e orientação do robô em campo, embora o filtro necessite de um processamento computacional pesado.

O sistema de fusão de sensores aplicada no trabalho pode ser visualizado na Figura 13, em que há a representação em um diagrama de blocos da fusão de sensores aplicada nos robôs da categoria MSL em Ismail, Purwanto e Arifin (2022).

A partir da Figura 13 é possível notar que o encoder e o giroscópio são utilizados para reconhecer a disposição do robô, sendo que são 2 encoders para detectar a translação e o giroscópio para detectar a rotação. Já a omnivisão detecta linhas brancas dentro do campo e, a partir das imagens cruas, há um processamento para gerar dados dessas linhas (i.e., cálculo da distância do robô até essas linhas por meio de regressão) antes de entrar no FP, que é ajudado por um bússola a fim de encontrar a posição .

A fusão de sensores atua combinando dados de posição e orientação do sistema de odometria e do FP. Os autores colocam que existem diferentes maneiras de realizar essa fusão (como comparação, junção, votação inteligente), mas no caso do trabalho são utilizados os

Figura 13 – Visão geral da fusão de sensores da odometria com o sistema de visão da categoria MSL.



Fonte: Retirado de Ismail, Purwanto e Arifin (2022)

métodos de fusão competitiva, onde os dados são tomados de forma independente, e complementar, em que há a utilização dos dados de ambos os sistemas de localização.

O teste realizado para validação do sistema de localização é o seguinte: o robô é movido manualmente para 20 coordenadas num campo de 9m x 6m, sendo que o movimento do robô é rastreado a fim de comparar com os dados de uma câmera global alocada em cima do campo. Depois de percorrer essas 20 coordenadas, são retirados a média, o desvio padrão do erro e o máximo erro dos sistemas de odometria, filtro de partículas e fusão de sensores.

A partir dos resultados, nota-se que no sistema de localização por odometria há um erro que cresce ao longo do tempo, enquanto para o FP e a fusão de sensores o erro não aumenta, enquanto a fusão dos dados acaba suavizando a saída quando comparado ao FP.

Além disso, um outro teste comparando ambos tipos de fusão de sensores foi realizado. O objetivo era movimentar o robô formando um zig-zag de maneira retangular. Os dados mostraram que o modo odometria é mais dominante, sendo que o sistema possui um tempo de resposta de aproximadamente 1.6ms para cada atualização de dados de localização.

Os autores concluem colocando que a fusão de sensores pode produzir dados ótimos ao combinar a odometria e a omnivisão com o FP. Os erros resultantes dos testes foram em x igual a  $10.5 \pm 7.8\text{cm}$ , em y igual a  $7.6 \pm 6.8\text{cm}$  e em  $\theta$  igual a  $1.9 \pm 1.2^\circ$ . Esses resultados são

melhores do que a utilização dos dados da odometria sozinhos, além de serem mais suaves do que o FP sozinho.

Este trabalho, por estar inserido na lógica do futebol de robôs, tem grande importância por utilizar um sistema de visão, mesmo que seja embarcado, diferente do sistema de visão da SSL. Além disso, o trabalho também compara duas maneiras de realizar a fusão de sensores com sensores que são semelhantes aos que foram utilizados nesse projeto.

Em Coito et al. (2014) é realizado um estudo de fusão de sensores utilizando um sistema de visão e IMU por meio de um filtro de Kalman de múltiplas taxas. No caso, a IMU possui um processamento bem mais rápido do que o sistema de visão, além de que a taxa de atualização da primeira é constante enquanto o da segunda costuma ter oscilações.

No caso, o robô utilizado é do tipo direcional, sendo 4 rodas que são unidas em pares por 2 eixos. A IMU é alocada no centro do robô, sendo que o giroscópio é alinhado com o eixo vertical.

Para lidar com as diferentes taxas de amostragem dos sistemas de visão e medição inercial é utilizado um KF discreto de múltiplas taxas, que segue a Equação (24), em que  $Z$  é o vetor de estados  $\begin{bmatrix} v_x & x & v_y & y & \theta \end{bmatrix}$ ,  $Y_k$  é o vetor de valores medidos,  $n_k$  e  $w_k$  são ruídos. Já  $A_k$  e  $G_k$  são as matrizes do sistema. No caso, as matrizes  $A_k$  e  $G_k$  são descritas conforme a Equação (25).

$$\begin{aligned} Z_{k+1} &= A_k Z_k + G_k n_k \\ Y_k &= C_k Z_k + w_k \end{aligned} \tag{24}$$

$$A_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \delta t_k & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \delta t_k & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (25)$$

$$G_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Os testes simulados foram realizados com o sistema inercial atualizando a uma taxa de 100Hz, enquanto o sistema de visão é simulado e fornece 2 medidas de posição por segundo. Os

resultados mostraram que a trajetória do robô não apresenta um erro crescente ao longo do tempo, permanecendo abaixo de 3cm na maior parte do teste, sendo que o desvio aumenta quando o robô realiza uma curva rápida com um raio pequeno, o que poderia ser mitigado com um sistema de visão mais rápido, segundo os autores.

Já nos resultados experimentais, o robô realizou uma trajetória com 4 seções em que se movia com velocidade linear de 10cm/s e uma velocidade angular de 0.2rad/s. As estimativas do filtro de Kalman eram calculadas a cada 25ms. Os resultados demonstraram uma boa estimativa da posição real sendo que, de acordo com os autores, a precisão e a taxa de amostragem se adequam para boa parte de aplicações de controle.

O trabalho traz informações importantes para fusão de sensores com diferentes taxas de atualização, principalmente por conta do uso de um sistema de visão, que normalmente possui um delay consideravelmente alto em relação aos outros sensores, como IMU e encoder. Os autores colocam que a cada momento de fusão o conjunto de sensores disponíveis varia, então a Equação (24) leva em conta a taxa de atualização.

Continuando com aplicações na competição do futebol de robôs, em Luis Aguiar et al. (2017) é feito um estudo comparando o filtro de Kalman estendido, o filtro de Kalman *unscented* (do inglês, UKF) e o filtro de Kalman linear para rastreamento de posição para a categoria IEEE *Very Small Size*, que é como uma categoria abaixo em relação a SSL quanto ao tamanho dos robôs, já que os robôs devem caber num cubo de 7.5cm de lado.

Os robôs da categoria, em suma, são diferenciais, que possuem 2 motores acoplados em 2 rodas normalmente. Os autores colocam que para o rastreamento da posição podem ser utilizados 2 modelos estocásticos para um robô desse tipo, cada um com uma representação de estados diferente.

O primeiro modelo é chamado de modelo *unicycle*, em que o estado é dado por  $x_k = \begin{bmatrix} x_k & y_k & \theta_k & v_k & \omega_k \end{bmatrix}^T$ . No caso, os três primeiros são as coordenadas e orientação de um ponto fixo entre as rodas do robô em relação a um *frame* de referência. No problema de rastreamento do robô oponente não se tem acesso às entradas de controle, mas a formulação do estado e a cinemática do robô são suficientes para extrair as estimativas da velocidade. Isso pode ser observado na Equação (26). É possível notar que as acelerações linear e angular são modeladas como um vetor aleatório Gaussiano com covariância  $Q$ .

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} v \times \cos \theta \\ v \times \sin \theta \\ \omega \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} w_a \\ w_\alpha \end{bmatrix} \quad (26)$$

Já o modelo de observação é dado pelo sistema de visão, também posicionado acima do campo, provendo observações de posição e orientação. Entretanto, esses dados são ruidosos, assumidos aditivos e Gaussiano com covariância  $R$ , como é possível ver na Equação (27).

$$z_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} + \begin{bmatrix} v_{x,k} \\ v_{y,k} \\ v_{\theta,k} \end{bmatrix} \quad (27)$$

O segundo modelo é chamado de dupla integração, que é demonstrado na Equação (28). No caso, o estado do robô é  $x_k = [x_k \ y_k \ v_{x,k} \ v_{y,k}]^T$  e o modelo de observação é obtido conforme a Equação (29). Na categoria VSS, pode não ser possível extrair de maneira fácil a orientação do oponente a partir do sistema de visão, já que o sistema de cores não é unificado, diferente da categoria SSL.

$$\begin{bmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ v_{x,k-1} \\ v_{y,k-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} w_{a,x} \\ w_{a,y} \end{bmatrix} \quad (28)$$

$$z_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} v_{x,k} \\ v_{y,k} \end{bmatrix} \quad (29)$$

Para validar as técnicas de rastreamento foram realizados testes com os robôs reais e na simulação. Os testes reais foram realizados em um campo da categoria VSS e foram extraídos as posições cartesianas e a orientação providas pelo sistema de visão.

Os dados reais mostraram que o EKF e o UKF obtiveram resultados iguais, indicando que não há necessidade da utilização de uma técnica mais complexa como a segunda, já que o modelo do VSS não é um sistema que representa grandes não-linearidades, portanto o EKF seria preferível por conta da simplicidade e custo computacional. Além disso, os autores colocam que com o modelo *unicycle* é possível alcançar uma estimativa mais completa do estado por incluir a orientação e velocidade angular do robô.

Já para a simulação, com o modelo de dupla integração utilizando o filtro de Kalman linear, apesar da baixa precisão para representar a dinâmica de um robô diferencial, os autores colocam que a performance de estimativa foi quase a mesma em relação ao EKF e UKF com o modelo *unicycle*, visto que o EKF apresentou um erro quadrático médio apenas 1.5% menor que o KF para estimativa de velocidade. Isso mostra também que o modelo consegue estimar com sucesso a velocidade no caso em que não se possui a orientação do robô.

Este trabalho é mais um inserido na realidade do futebol de robôs e numa categoria tão dinâmica quanto a SSL. Além disso, o autor mostra duas maneiras possíveis para representar o modelo dinâmico de um sistema, mesmo que os robôs dessa categoria sejam diferenciais.

Em Pacheco, Silva e Farrell (2023) é feito um estudo para atingir um posicionamento acurado de robôs inteligentes para agricultura por meio da integração de um GPS e sensores de odometria de baixo custo, justamente para que seja acessível aos pequenos e médios agricultores, utilizando o EKF para conseguir essa integração.

Para realizar a etapa de predição do EKF os autores definem os estados como sendo os erros na solução de navegação utilizando odometria. No caso, os estados são o erro na orientação do robô pela odometria, os erros nas medições da posição norte e leste da odometria, o desvio associado ao raio das rodas direita e esquerda do robô, o erro no tempo de recepção do sinal do GPS e o desvio do GPS. O processo do modelo é estabelecido pelas equações que governam a evolução dos erros de estado da odometria ao longo do tempo.

Já a etapa de atualização é dividida a partir da topologia utilizada. Como nesse trabalho não haverá a utilização do sensor GPS como atualização do EKF, dessa etapa basta falar que na topologia fracamente acoplada a inovação da medição é determinada como a diferença entre a posição cartesiana fixa e centrada na Terra medida pelo GPS e a estimativa correspondente da odometria. Já na topologia fortemente acoplada a inovação da medição é calculada como a diferença entre as pseudo-distâncias medidas pela antena receptora do GPS para todos satélites em vista e a solução de navegação odométrica.

O trabalho demonstrou bons resultados, com uma melhoria significativa da precisão e acurácia do robô no que diz respeito às estimativas de posição e orientação, com melhorias que podem chegar à 97% comparadas com os dados tanto da navegação utilizando odometria somente quanto os dados utilizando só a navegação por GPS.

Os autores concluem que, apesar dos resultados positivos alcançados no trabalho, o sistema de navegação odométrica auxiliada por GPS não alcançou as especificações de acurácia de posicionamento exigidas pela norma SAE J2945 (SAE INTERNATIONAL, 2016).

Apesar dos resultados não serem satisfatórios para os autores por não cumprirem o determinado pela norma especificada, o modo que a etapa de predição é descrita no trabalho é uma ideia diferente que pode ser utilizada neste trabalho.

Já em Dyer et al. (2020) é realizado o teste de uma estratégia diferente para utilização de filtros em um robô omnidirecional com 4 rodas. No caso, é utilizada uma estratégia multi-filtro em que um KF estima a velocidade da roda a partir da dinâmica do motor e tem como medidas a leitura de velocidade de um encoder e o valor de corrente no motor, além de um EKF e um filtro suave de estrutura variável (FSEV) para estimativa do estado do robô.

Os testes conduzidos buscaram realizar diferentes combinações e encontrar a melhor, isto é, utilizando ou não o KF e, em seguida, utilizando ou não um dos outros 2 filtros para estimativa do robô. As simulações foram conduzidas utilizando MATLAB, apesar de que as constantes físicas do motor foram conseguidas experimentalmente por meio de um osciloscópio, um multímetro e um gerador de ondas.

Os resultados mostraram que apenas a utilização do KF nas rodas melhorou a precisão da estimativa em relação a utilização de nenhum filtro (erro médio quadrático de 10.02cm para 9.28cm no eixo X, por exemplo). Já a utilização de um filtro para estimativa do robô, mesmo sem um filtro para as rodas, acabou dobrando a precisão da estimativa, sendo que o FSEV se mostrou ligeiramente mais acurado (4.63cm de erro médio quadrático para ambos no eixo X). Já a utilização de ambos em conjunto aumenta a precisão duplamente, gerando um erro médio quadrático menor que 2cm.

O trabalho demonstra a qualidade da utilização do filtro de Kalman para predição do estado do robô, além de mostrar uma maneira interessante para estimar a velocidade das rodas do veículo a partir do modelo do motor, mostrando que a junção de ambos traz resultados melhores estimando o estado atual.

A partir dos trabalhos observados, é notório que há uma lacuna no que diz respeito a comparação dos sensores normalmente utilizados em projetos de robótica para sistemas de localização, principalmente em aplicações relacionadas a futebol de robôs, sendo os únicos descritos nessa Seção os trabalhos de Luis Aguiar et al. (2017) e também (ISMAIL; PURWANTO; ARIFIN, 2022), que descrevem o desenvolvimento de um sistema de localização baseado em fusão de sensores, mas fazem outras comparações, como o método de fusão, ao invés de comparar diferentes sensores.

## 5 METODOLOGIA

Neste capítulo serão apresentados o domínio de teste do projeto proposto, os testes a serem realizados, como os sensores foram combinados, a calibração dos sensores e quais métricas foram analisadas para avaliação dos testes propostos.

### 5.1 DOMÍNIO DE TESTES

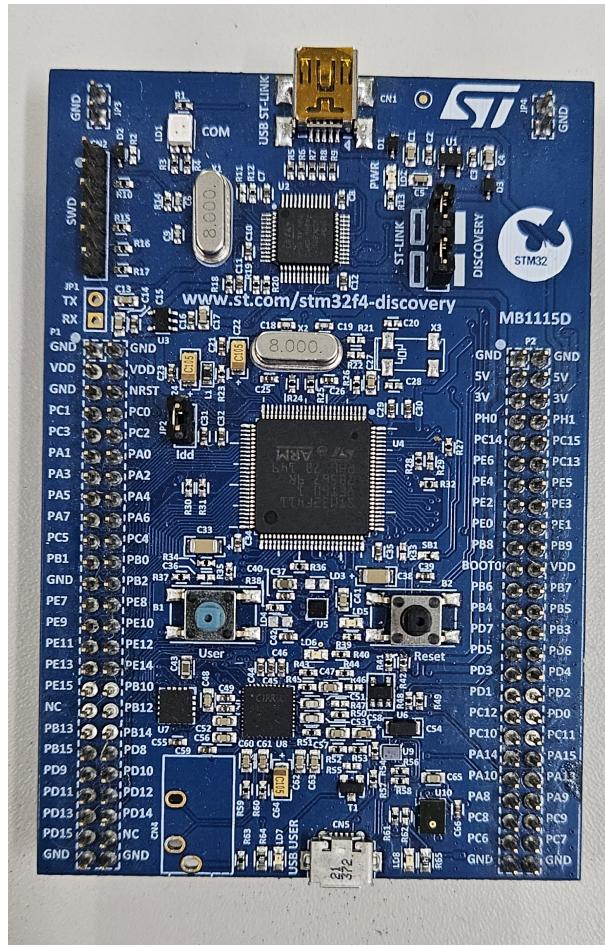
Neste trabalho foi utilizado o domínio de teste da categoria SSL de futebol de robôs da RoboCup (ROBOCUP, 2020), descrito na Seção 2.1. Nele, as combinações de sensores foram testadas e comparadas. Para a realização dos testes propostos, foi utilizado o campo de futebol do laboratório da equipe RoboFEI, que possui 4,3 X 3,6m, o que possibilita uma boa estimativa da movimentação do robô em um campo oficial da competição.

O que torna a SSL um interessante campo de teste para as combinações dos sensores são algumas características dos robôs e das partidas, como:

- a) **Velocidade alta dos robôs:** A dinamicidade das partidas se dá muito por conta da alta velocidade atingida pelos robôs, sendo necessário um bom sistema de controle de posição, o que requer um estimador de posição utilizando sensores além das câmeras da liga;
- b) **Altas penalidades para colisões:** as regras da competição punem severamente os times que causam muitas colisões, podendo levar a diversos cartões amarelos (um robô deve ficar fora do campo por 2 minutos), e até cartões vermelhos (o robô é expulso pelo resto da partida);
- c) **Alta precisão necessária para realização de jogadas:** por conta do pequeno tamanho tanto dos robôs e da bolinha, para realizar jogadas rápidas que evitem a chegada de robôs adversários, faz-se necessário uma alta precisão de posicionamento dos robôs.

Sobre o robô utilizado na equipe RoboFEI, os sensores giroscópio e acelerômetro estão presentes na placa de desenvolvimento STM32F411E-Disco (STMICROELECTRONICS, 2020), que é a placa onde está alocado o microcontrolador do robô. É possível observar uma imagem deste componente na Figura 14. Esta placa fica alocada numa placa eletrônica chamada de principal, que também contém os *drivers* de ativação dos motores do robô e o módulo de comunicação via rádio frequência.

Figura 14 – Placa de desenvolvimento STM32F411E-Disco utilizada.



Fonte: Autor

## 5.2 COMBINAÇÃO DOS SENSORES

Nesta seção serão definidos os sensores a serem utilizados e como eles foram combinados para realizar a predição e correção do filtro de Kalman.

Os sensores utilizados nos testes são os descritos ao longo da Seção 2.2, que são: IMU (Giroscópio + Acelerômetro), encoders, sistema de câmeras utilizado na SSL, além do modelo cinemático do robô omnidirecional (apresentado na Seção 2.1). Eles foram combinados entre as etapas de predição e correção a fim de encontrar a melhor configuração para realizar a estimativa de posição utilizando Filtro de Kalman Estendido, como é descrito na Tabela 1.

No caso, os três primeiros cenários foram realizados utilizando o modelo cinemático do sistema na fase de predição, variando qual sensor era utilizado na fase de correção. Estes primeiros cenários serviram como uma introdução para o desenvolvimento do filtro de Kalman. No caso, o modelo cinemático do sistema leva em conta o comando de velocidade enviado para

Tabela 1 – Combinações dos sensores na predição e correção do Filtro de Kalman.

Teste	Predição	Correção
Cenário 1	Modelo do sistema	Câmera
Cenário 2	Modelo do sistema	Encoders
Cenário 3	Modelo do sistema	IMU
Cenário 4	IMU	Câmera
Cenário 5	IMU	Encoders
Cenário 6	Encoders	Câmera
Cenário 7	Encoders	IMU
Cenário 8	-	Câmeras

Fonte: o Autor

o robô da categoria SSL (ver Equação (1) na Seção 2.1). Os dois próximos cenários foram realizados com a IMU na fase de predição, enquanto na fase de correção foram utilizados os outros dois sensores, respectivamente. Os dois cenários seguintes tiveram a mesma ideia dos testes 4 e 5, mas na fase de predição foram utilizados os encoders e na fase de correção os outros dois sensores. Já o cenário 8 é o sistema de localização atual da equipe, que utiliza apenas os dados vindos do sistema de câmeras, ou seja, não há uma fusão de sensores propriamente dita, e serve como comparação para os outros cenários.

### 5.3 IMPLEMENTAÇÃO

Esta seção descreve a visão geral da implementação dos algoritmos deste trabalho, isto é, a linguagem utilizada para programá-los, o hardware utilizado e o fluxo geral do sistema de localização proposto.

Em relação à linguagem de programação, para codificar os algoritmos de fusão de sensores no microcontrolador descrito na Seção 5.1 foi utilizada linguagem C. A biblioteca CMSIS-DSP foi usada principalmente para auxiliar nos cálculos matriciais necessários nas equações do KF, assim como para auxílio em cálculos que envolvam trigonometria (ARM, 2024).

O hardware utilizado no trabalho pode ser dividido em dois: o que é interno ao robô, e o que é externo ao robô. Internamente ao robô têm-se os sensores, que foram descritos na Seção 2.2, além do microcontrolador utilizado para cálculo de fusão de sensores para localização de maneira embarcada. De modo externo ao robô deve-se considerar o notebook que é utilizado tanto para recebimento dos dados vindos do sistema de câmeras da categoria, mas também para

cálculo da trajetória do robô em campo e para registro de dados a fim de análise dos resultados obtidos com os algoritmos. As especificações técnicas do notebook estão descritas na Tabela 2.

Tabela 2 – Especificações de hardware do notebook.

Sistema Operacional	Ubuntu 22.04.5 LTS x86_64
CPU	Intel i7-7700HQ CPU @ 2.80GHz
GPU	NVIDIA Geforce MX150
Memória RAM	16GB

Fonte: o Autor

Além disso, as câmeras utilizadas pelo SSL-Vision para detecção dos robôs utilizando o padrão colorido em cima deles são as seguintes:

- a) Stingray F046C (TECHNOLOGIES, 2011) + Lente Tamron 12VM412ASIR
- b) Logitech BRIO 4K UHD (LOGITECH, 2021)

Durante os testes, a transferência dos dados processados pelo SSL-Vision para o robô é feita via rádio-frequência, que a equipe já utiliza para enviar outros comandos para o robô, como velocidades locais, ativação do sistema de chute e acionamento do sistema de drible. Além disso, o *feedback* das informações do robô para avaliação das combinações e dos sensores é transmitida a partir do robô para o computador central via rádio-frequência.

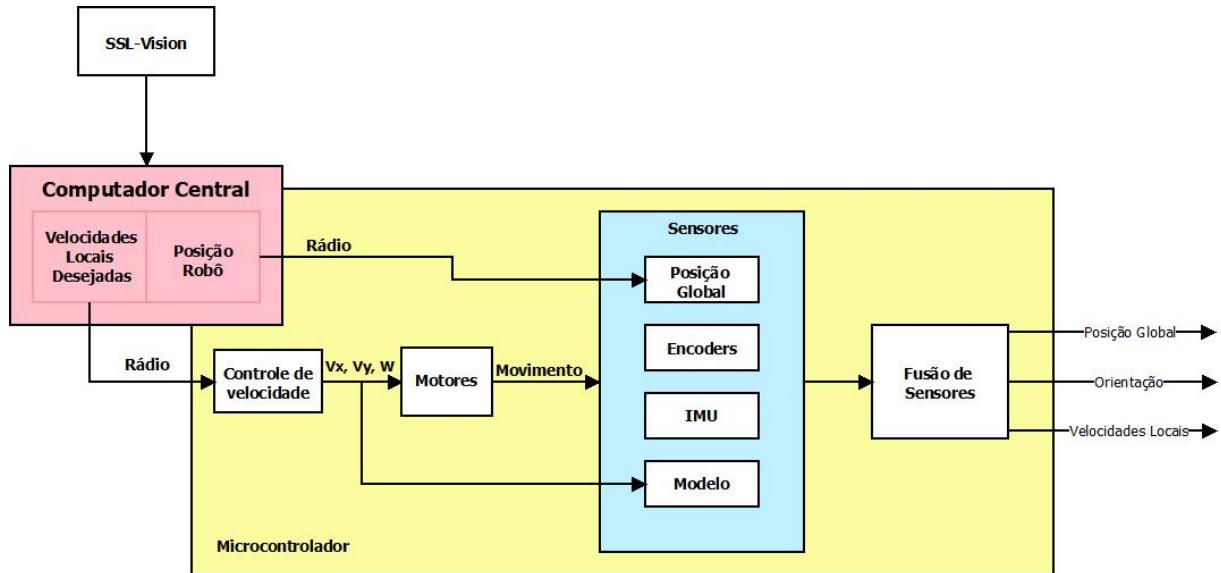
O transceptor utilizado para comunicação entre a estação do rádio conectada ao computador da equipe e o robô é o módulo nRF24L01, que no caso opera numa frequência de 2.4GHz e necessita de poucos componentes para ser projetado como um sistema de rádio (NORDIC SEMICONDUCTOR, 2007). O módulo é configurado e operado via SPI acessando o mapa de registradores que servem para configurar o componente.

O computador que roda a estratégia da equipe faz a formatação do pacote a ser enviado via rádio para o robô, sendo que o pacote pode ter um comprimento máximo de 32 bytes por conta da limitação do próprio nRF24L01. Assim, o pacote é transmitido para o transceptor por meio de comunicação serial via USB e, em seguida, o pacote é enviado para o robô, que verifica se o pacote pertence a ele pelo ID que é transmitido. O tamanho do pacote acaba sendo um gargalo tanto para o envio quanto para o recebimento das informações, visto que há outras informações que são transmitidas via rádio, como as velocidades linear, angular e do sistema de drible, ID do robô e tipo de chute (já que o robô pode fazer chutes retos ou "cavadinhas"), além da posição do robô em campo.

As posições lidas e transmitidas para os robôs vindas do sistema global de câmeras da liga necessitam de 2 bytes cada, totalizando assim 6 bytes para transmissão dessas informações ao robô. Isto porque apenas 1 byte não é suficiente para transferir esses dados, já que seriam apenas 256 possibilidades ( $2^8$  pela quantidade de bits num byte) e, por exemplo, a posição linear em X do robô pode ser de  $\pm 4500\text{mm}$ , totalizando 9000 posições, logo 65536 possibilidades ( $2^{16}$  utilizando 2 bytes) já é o suficiente.

O diagrama geral do funcionamento do sistema de localização por fusão de sensores de forma embarcada ao robô pode ser observada na Figura 15. Nela, é possível entender de maneira clara a diferença entre o que é calculado no computador central da equipe (recebimento dos dados do SSL-Vision e cálculo das velocidades locais desejadas) e o que é feito no microcontrolador do robô (recebimento dos dados do computador central, implementação dos sensores e cálculo da fusão de sensores).

Figura 15 – Diagrama de implementação da fusão de sensores para localização.



Fonte: Autor

#### 5.4 GROUND TRUTH

O *ground truth* é parte determinante desse projeto, visto a necessidade de comparar o sistema de localização desenvolvido com a posição real do robô em campo. Por conta disso, um sistema para calcular a posição real do robô foi desenvolvido utilizando dois LiDARs a fim de encontrar o formato arredondado dos robôs da categoria SSL.

A Seção 6.1 traz o método desenvolvido em detalhes, assim como os resultados obtidos em comparação ao sistema de câmeras da categoria SSL, mas também em relação à medição manual utilizando uma trena.

O sistema roda em conjunto ao recebimento dos dados e ao cálculo de trajetória do robô no notebook descrito na Tabela 2. Para performar a detecção, foram utilizados os lasers RPLIDAR A1 (SLAMTEC, 2016) e RPLIDAR S2 (SLAMTEC, 2021), os modos de operação deles estão descritos na Tabela 3. Além disso, ambos os sensores operaram com frequência de 10Hz.

Tabela 3 – Especificações dos LiDARs utilizados.

Sensor	Princípio	Modo de operação	Pontos capturados
RPLIDAR A1	Triangulação	Boost	787
RPLIDAR S2	Time-of-Flight	Standard	1612

Fonte: o Autor

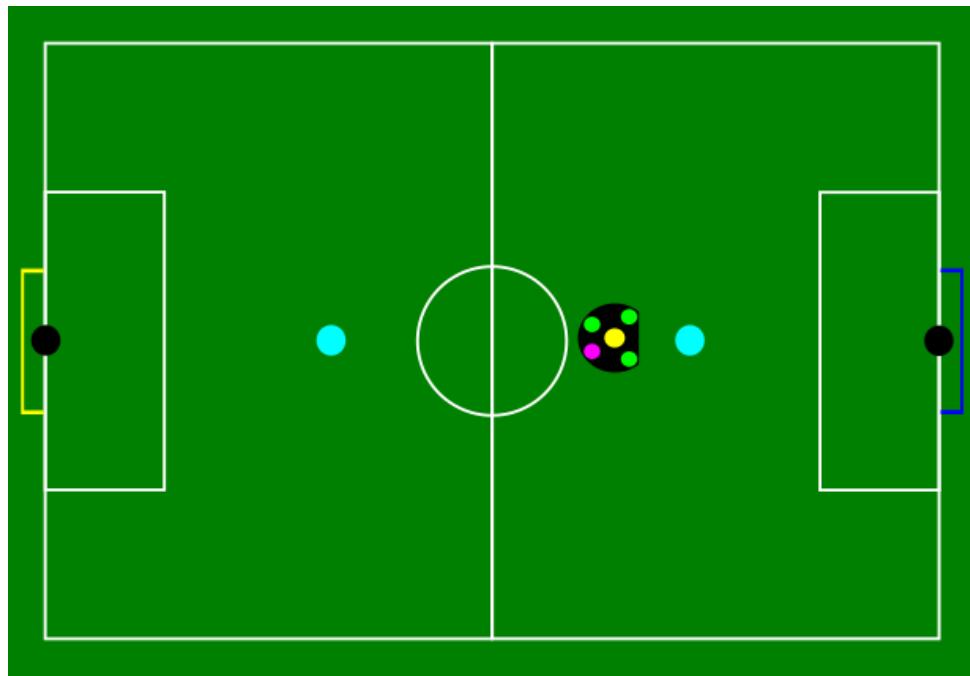
Os LiDARs foram posicionados no meio de cada um dos gols a fim de obter uma melhor detecção, visto que o laser A1 tem um alcance máximo de 6 metros, e o erro de detecção aumenta gradualmente com a distância. Uma representação da posição dos lasers (pontos pretos), posição das câmeras (pontos azuis) e do robô pode ser vista na Figura 16.

Apesar de os LiDARs estarem posicionados aproximadamente numa mesma altura, é importante ressaltar que por conta da modulação da frequência do sinal enviado por cada um deles durante a varredura realizada, além do próprio funcionamento rotativo da varredura, que faz a janela de interferência ser uma fração de segundo, não há interferência entre ambos os LiDARs.

## 5.5 SENSORES INERCIAIS

Diferente dos dados das câmeras, os dados dos outros sensores são todos processados no microcontrolador embarcado no robô. Como dito nas Seções 2.2.4 e 2.2.5, o acelerômetro e o giroscópio já estão presentes na placa de desenvolvimento utilizada no hardware do robô da equipe RoboFEI (Figura 14), o que facilita para aquisição dos dados do sensor.

Figura 16 – Representação do posicionamento dos lasers em campo.



Fonte: Autor

### 5.5.1 Acelerômetro

Como colocado na Seção 2.2.5, os registradores do componente LSM303AGR podem ser acessados por meio das interfaces seriais I<sup>2</sup>C ou SPI. No caso deste projeto, essa comunicação é realizada via I<sup>2</sup>C, onde o LSM303AGR é o escravo e o microcontrolador é o mestre.

A biblioteca para obtenção dos dados do acelerômetro é disponibilizada pela própria desenvolvedora (STMICROELECTRONICS, 2023b). Sua utilização facilita a aquisição de dados, visto que a classe já possui métodos para inicialização do componente, filtragem passa-alta e leitura dos valores de aceleração do sensor, sendo necessária a escolha de alguns parâmetros de configuração, como frequência de saída de dados, escala e resolução.

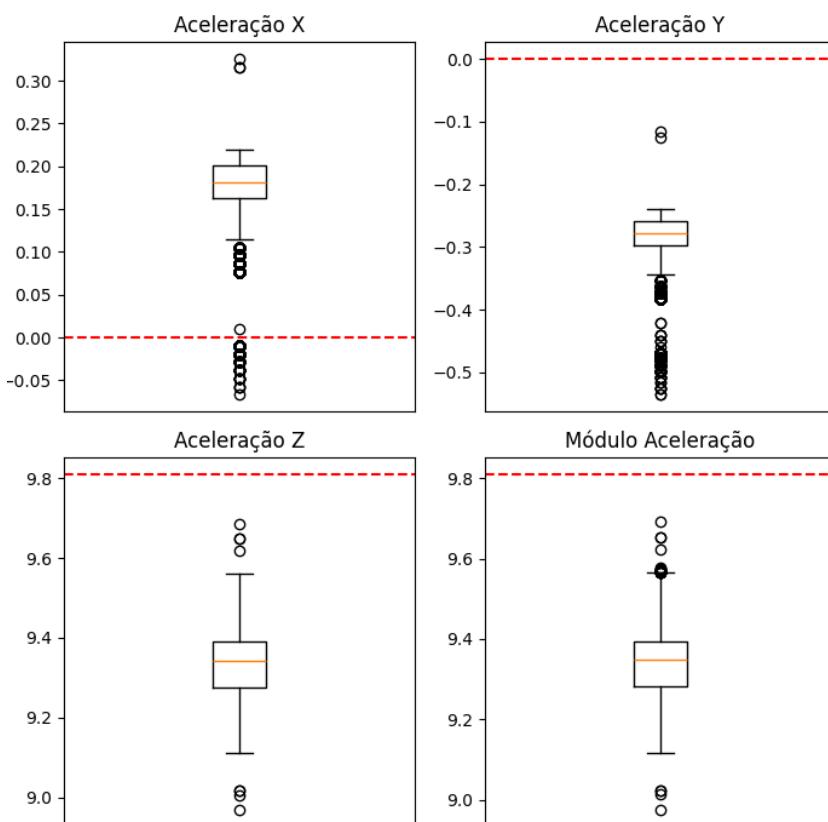
O sensor foi configurado com uma frequência de saída de 400Hz para que os dados a serem utilizados para estimativa de posição sejam atualizados de forma rápida. Já a escala escolhida, que define os limites do sensor, foi de  $\pm 2g$ , já que as acelerações de um robô da categoria SSL não são maiores do que esse limite. Além disso, o sensor foi operado no modo de energia normal, o que permite uma saída de dados de até 10 bits. O componente possui outros dois modos, o de baixo consumo que produz uma saída de até 8 bits, e o de alta resolução, que

produz uma saída de até 12 bits, porém a questão energética e/ou da resolução não são gargalos do projeto.

### 5.5.1.1 Calibração

Após a configuração do acelerômetro conforme explicação anterior, decidiu-se colher algumas amostras de dados do sensor. A Figura 17 apresenta os dados de 10000 amostras adquiridas do acelerômetro numa mesa plana com a placa em repouso, no caso foram extraídos dados dos três eixos, além do valor do módulo da aceleração, que no caso deve ser o valor da gravidade. O eixo vertical do gráfico representa a aceleração em  $m/s^2$ . Na Tabela 4 é possível observar alguns outros dados estatísticos das amostras do acelerômetro antes da calibração do sensor.

Figura 17 – 10000 amostras dos dados do acelerômetro antes da calibração.



Fonte: Autor

Legenda: o tracejado em vermelho representa a medida esperada em cada um dos eixos e também do módulo da aceleração.

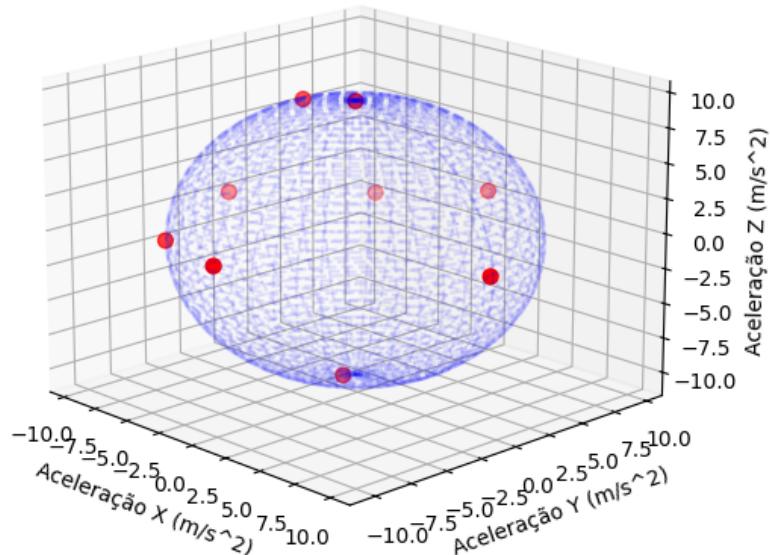
Tabela 4 – Análise estatística dos dados do acelerômetro antes da calibração.

Dados	Média	Desvio Padrão ( $\sigma$ )	Variância ( $\sigma^2$ )
Aceleração X	0.169561	0.054271	0.002945
Aceleração Y	-0.294967	0.060317	0.003638
Aceleração Z	9.327501	0.091915	0.008450

A partir da Figura 17, é nítido que as amostras estão longe das medidas esperadas, como uma diferença de quase  $0.5m/s^2$  no módulo geral, por exemplo. Além disso, a partir da Tabela 4 é possível notar também a média do valor da aceleração em Z, que é um valor bem distante do esperado. Essa grande diferença invalida a utilização do acelerômetro, mostrando a necessidade de uma calibração prévia do sensor.

A calibração do acelerômetro, como descrito na Seção 2.2.7.1, é realizada pelo chamado método da esfera e é necessária a aquisição de 4000 amostras do sensor em 9 posições diferentes, sendo que em cada uma dessas posições é realizada a média das amostras tomadas, o que no caso são pontos utilizados para encontrar os desvios do acelerômetro. A Figura 18 mostra a esfera corrigida cujo raio é o valor da gravidade e o centro são os desvios do sensor, enquanto os pontos são as médias de cada uma das posições de calibração.

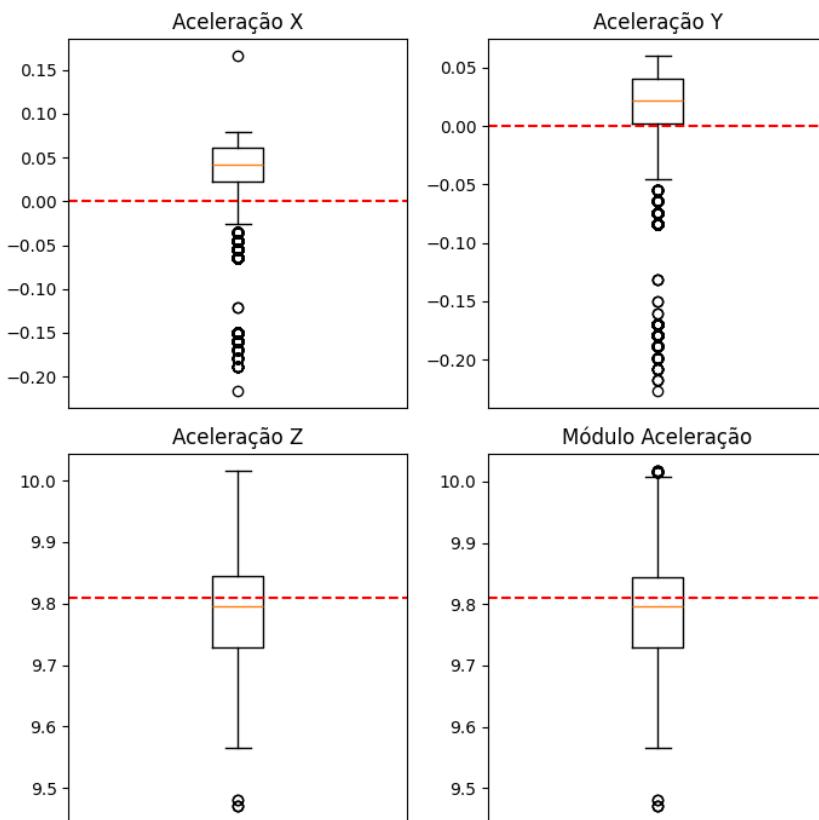
Figura 18 – Esfera de calibração do acelerômetro.



Fonte: Autor

Nota-se que os pontos estão bem em cima da esfera, o que demonstra êxito da calibração. Além do mais, na Figura 19 é possível observar os dados dos 3 eixos e do módulo desses valores das 10000 amostras (o eixo vertical do gráfico representa a aceleração em  $m/s^2$ ) captadas depois da calibração e realizadas nas mesmas condições em que as amostras antes da calibração foram captadas. Além do mais, na Tabela 5 é possível notar algumas outras medidas das amostras colhidas depois da calibração.

Figura 19 – 10000 amostras dos dados do acelerômetro depois da calibração.



Fonte: Autor

Legenda: o tracejado em vermelho representa a medida esperada em cada um dos eixos e também do módulo da aceleração.

Tabela 5 – Análise estatística dos dados do acelerômetro depois da calibração.

Dados	Média	Desvio Padrão ( $\sigma$ )	Variância ( $\sigma^2$ )
Aceleração X	0.027601	0.055726	0.003105
Aceleração Y	0.006058	0.057984	0.003362
Aceleração Z	9.784110	0.090803	0.008245

Os dados tanto da Figura 19 quanto da Tabela 5 mostram o êxito da calibração do sensor, visto que as médias das amostras estão bem mais coerentes do que seria o esperado numa superfície plana. Apesar disso, é possível notar que os dados do acelerômetro possuem uma amplitude consideravelmente grande, com diversos dados considerados discrepantes, que são os círculos fora dos quartis dos gráficos de caixa.

### **5.5.2 Giroscópio**

Assim como o acelerômetro, o giroscópio também já está presente na placa de desenvolvimento utilizada no hardware da equipe, o que facilita sua utilização e, no geral, segue a mesma ideia para aquisição de dados, já que também há uma biblioteca disponibilizada pela própria fabricante (STMICROELECTRONICS, 2023a).

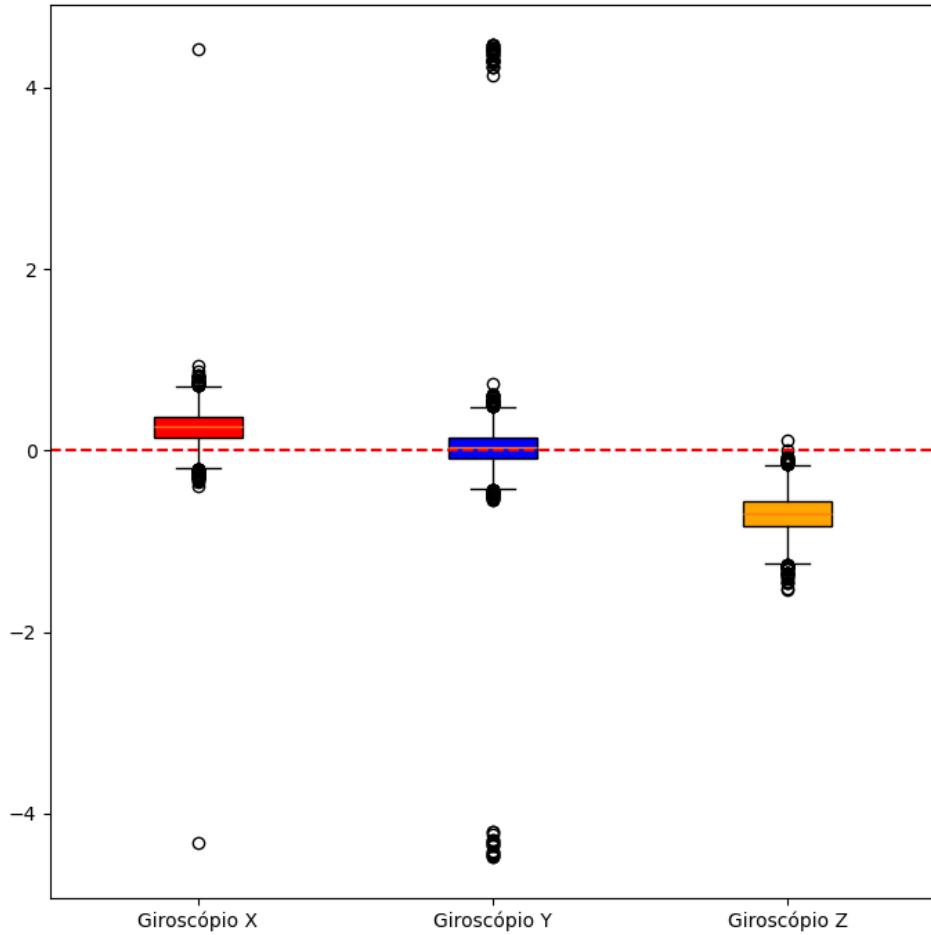
Como colocado na Seção 2.2.4, o sensor I3G4250D é capaz de disponibilizar os dados de velocidade angular nos três eixos a partir de uma interface digital SPI, assim como também há uma interface I<sup>2</sup>C compatível disponível. No caso desse projeto, o protocolo de comunicação escolhido foi o SPI para o giroscópio, que será o escravo no caso dessa comunicação em relação ao microcontrolador, que é o mestre.

O sensor foi configurado com uma frequência de saída de 400Hz a fim de que os dados a serem utilizados pelo filtro de Kalman para estimativa de posição estejam disponíveis numa taxa rápida. A escala escolhida foi de 500 graus por segundo (do inglês, dps), que é o limite que os dados podem alcançar, e esse valor foi escolhido visto que um robô da categoria SSL pode alcançar uma velocidade angular de mais de 1 rotação por segundo, mas também pelo fato de a escala de um sensor ser uma troca entre precisão e detecção de altos valores, visto que quanto menor a escala melhor a precisão dos dados. No caso desse sensor, o seu único modo de operação é o normal.

#### **5.5.2.1 Calibração**

Tal como aconteceu com os dados do acelerômetro, decidiu-se extrair amostras de dados do giroscópio. A Figura 20 apresenta os dados de 1000 amostras adquiridas do giroscópio numa mesa plana com a placa em repouso, no caso foram extraídos dados dos três eixos. O eixo vertical do gráfico representa a velocidade angular em °/s. Já a Tabela 6 mostra algumas estatísticas das 10000 amostras captadas.

Figura 20 – 10000 amostras dos dados do giroscópio antes da calibração.



Fonte: Autor

Legenda: o tracejado em vermelho representa a medida esperada em cada um dos eixos.

Tabela 6 – Análise estatística dos dados do giroscópio antes da calibração.

Dados	Média	Desvio Padrão ( $\sigma$ )	Variância ( $\sigma^2$ )
Velocidade Angular X	0.255584	0.176988	0.031325
Velocidade Angular Y	0.036696	0.357976	0.128147
Velocidade Angular Z	-0.700549	0.203428	0.041383

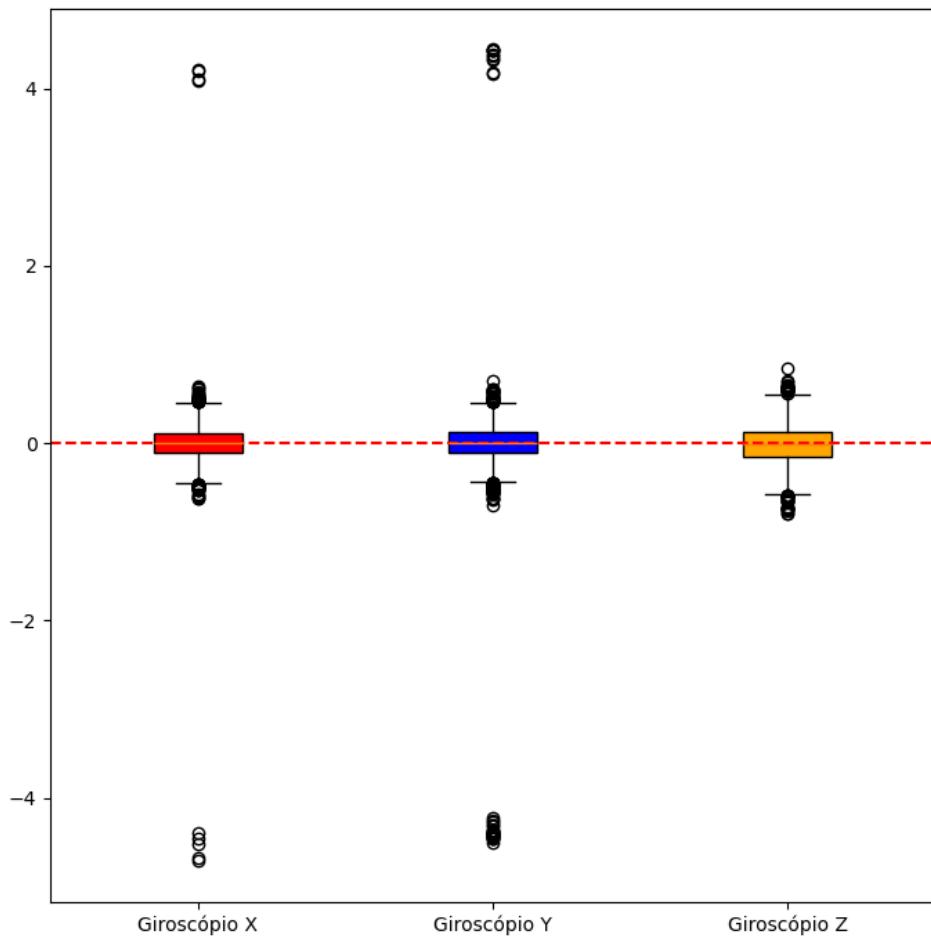
Assim como aconteceu com o acelerômetro, os dados mostram a necessidade da calibração do giroscópio por conta da diferença significante entre os dados coletados e as medidas esperadas. Essa grande diferença invalida a utilização do sensor, como no caso da velocidade angular em torno do eixo Z, no qual a mediana dos dados está perto de  $1^\circ/s$ .

A calibração do giroscópio, como descrito na Seção 2.2.7.2, é feita seguindo a mesma ideia da calibração do acelerômetro, sendo que no caso do primeiro são tomadas 30000 amostras

contendo dados dos 3 eixos em uma única posição, diferente do segundo que necessitava de diferentes posições para uma maior precisão.

A Figura 21 demonstra a correção realizada nos dados do giroscópio, visto que a mediana dos dados nos 3 eixos em 10000 amostras estão mais próximas das medidas esperadas, o que é corroborado ao observar média dos valores na Tabela 7. O eixo vertical do gráfico representa a velocidade angular em  $^{\circ}/s$ .

Figura 21 – 10000 amostras dos dados do giroscópio depois da calibração.



Fonte: Autor

Legenda: o tracejado em vermelho representa a medida esperada em cada um dos eixos.

Apesar do êxito da calibração dos sensores, tal como aconteceu com o acelerômetro, é possível notar que existe uma certa amplitude nos dados dos giroscópio ao observar o gráfico da Figura 21 em que há muitos valores discrepantes, que são os círculos ao longo do eixo vertical.

Tabela 7 – Análise estatística dos dados do giroscópio depois da calibração.

Dados	Média	Desvio Padrão ( $\sigma$ )	Variância ( $\sigma^2$ )
Velocidade Angular X	0.001566	0.210595	0.044350
Velocidade Angular Y	-0.004188	0.318366	0.101357
Velocidade Angular Z	-0.011643	0.204092	0.041654

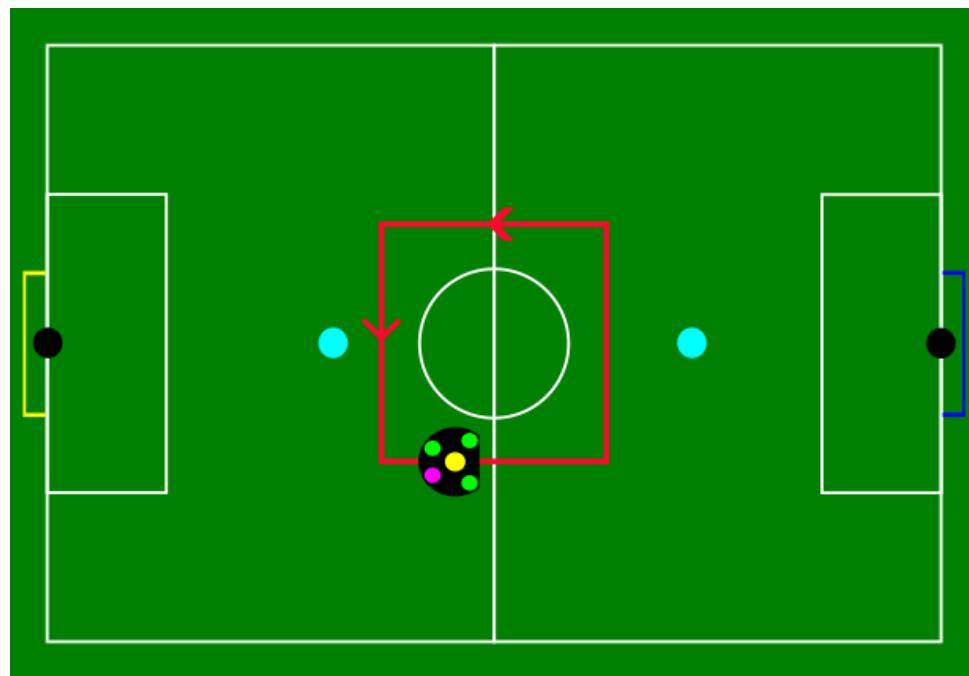
## 5.6 TESTES

Nesta seção serão definidos os testes que foram realizados a fim de obter dados relevantes para análise posterior das técnicas e combinações de sensores implementadas neste trabalho.

A fim de verificar qual a melhor combinação de sensores para realização da estimativa de posição utilizando Filtro de Kalman Estendido de robôs da SSL, além de pontos fortes e fracos de cada combinação, foram realizados testes em dois cenários.

O primeiro teste é um trajeto em forma de quadrado localizado na região central do campo, como é mostrado na Figura 22, com 70cm de lado. Como o trajeto está centralizado no campo, foi possível analisar os resultados das diferentes combinações numa região em que há sobreposição das duas câmeras utilizadas, que no caso gera um problema de o mesmo objeto ter diferentes posições por ser visto por câmeras distintas, o que é demonstrado na Figura 23.

Figura 22 – Ilustração do teste de quadrado menor.



Fonte: o Autor

Figura 23 – Problema de sobreposição de câmeras.



Fonte: Retirado de Pauli e Tonidandel (2019)

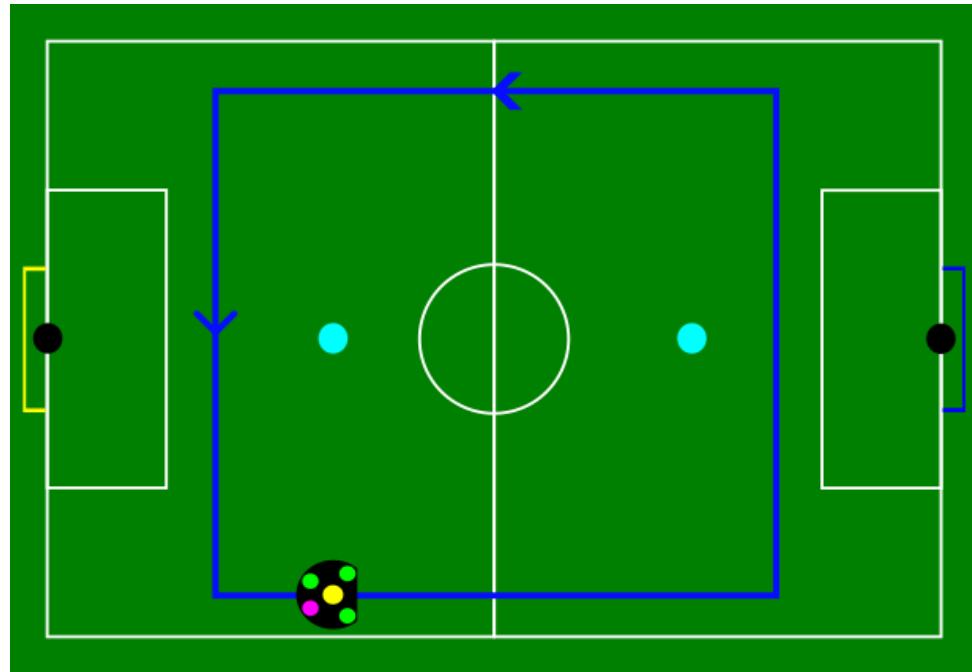
O segundo teste é um trajeto em forma de quadrado também, mas com um tamanho maior (150cm de lado) e passando por regiões do campo localizadas mais para as bordas, como é mostrado na Figura 24. Este trajeto tem o objetivo de analisar as diferentes combinações propostas de sensores em regiões localizadas nas bordas do campo, que apresentam problemas de distorção maiores causadas pela utilização de lentes olho de peixe nas câmeras. A Figura 25 mostra uma imagem que exemplifica essa distorção problemática nas bordas da imagem e, consequentemente, do campo.

Os testes foram realizados com velocidade linear e angular máximas de  $50\text{cm/s}$  e  $2\pi\text{rad/s}$ , respectivamente. É importante colocar também que, por conta do sistema de trajetórias da equipe RoboFEI o robô sempre segue o caminho proposto com a frente apontada para o vetor do trajeto, tal como demonstrado na Figuras 22 e 24.

## 5.7 AVALIAÇÃO

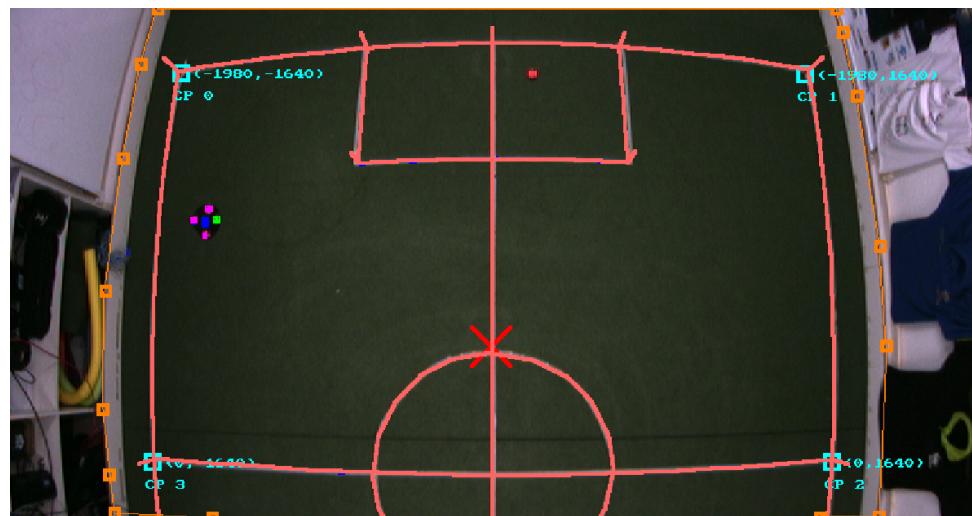
Nesta seção serão descritos os critérios de avaliação que foram utilizados para validar as diferentes combinações de sensores a fim de encontrar a melhor entre elas para realizar a

Figura 24 – Ilustração do teste de quadrado maior.



Fonte: o Autor

Figura 25 – Problema da distorção da imagem de câmeras com lente olho de peixe.



Fonte: o Autor

estimativa de posição de um robô móvel, além de analisar os pontos positivos e negativos das combinações.

A partir do feedback via rádio-frequência das posições calculadas pelo sistema de fusão de sensores embarcado, foi analisada a distância média do que é calculado de forma embarcada

com o sistema de *ground truth* desenvolvido. A Equação (30) demonstra o cálculo do Erro Quadrático Médio (RMSE, do inglês) da distância entre a posição calculada pelo sistema de localização utilizando fusão de sensores ( $\delta_{EKF,i}$ ) e a posição definida pelo sistema de *ground truth* ( $\delta_{GTS,i}$ ) na amostra  $i$  de um total de  $n$  amostras.

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (\delta_{EKF,i} - \delta_{GTS,i})^2} \quad (30)$$

Além da distância média durante os testes, também foi analisado o tempo de atualização das combinações. No caso, o tempo medido é o necessário para a realização dos cálculos para obtenção de uma nova estimativa de posição e velocidade do robô.

A fim de aumentar a confiabilidade dos dados a serem extraídos, cada teste foi realizado 10 vezes. A partir deles, foram extraídos a média, desvio padrão e o valor máximo de cada uma das métricas para uma análise concisa. Em cada um dos testes foram obtidas, em média, 700 amostras no trajeto quadrado menor e 1500 amostras no trajeto quadrado maior, quantidade suficiente para análise das combinações de sensores propostas.

As métricas foram escolhidas após estudos realizados nos trabalhos relacionados na área de localização de robôs móveis, sendo que a métrica do erro médio da posição estimada comparada com um *ground truth* é utilizada em todos, já o tempo de atualização é importante sua análise essencialmente pela aplicação do projeto na liga de futebol de robôs SSL da RoboCup. A partir dessas métricas é possível avaliar a qualidade dos diferentes sensores combinados quanto a qualidade da predição de posição do robô, assim como analisar o custo computacional, sendo que o melhor sistema de localização para um robô da categoria SSL possui uma posição predita com o menor erro médio possível, assim como o menor tempo de atualização de estimativas.

## 6 RESULTADOS

Nesta seção serão apresentados os resultados obtidos neste projeto através dos experimentos descritos na Seção 5.6. Além disso, também será descrita como foi feita a aquisição do *ground truth*, que serviu para comparação dos algoritmos. A fim de melhor entendimento, os resultados serão analisados graficamente ou por meio de tabelas.

### 6.1 AQUISIÇÃO DO GROUND TRUTH

Como descrito na Seção 5.4, foi necessário o desenvolvimento de um sistema utilizando 2 LiDARs externos ao robô para obter o *ground truth* do posicionamento do posicionamento do robô em campo a fim de realizar a comparação dos sistema de localização desenvolvido nesse projeto com a posição real do robô.

Embora o SSL-Vision seja utilizado como *ground truth* em diversos trabalhos (PAULI; TONIDANDEL, 2019; MELO; BARROS, 2022), nesse trabalho não é possível pelo sistema de visão global da liga ser um dos sensores considerados para comparação. Portanto, foi necessário criar um sistema de *ground truth* com LiDARs externos pela necessidade de obter a posição com o robô em movimento, embora seja possível obter o *ground truth* da posição manualmente com o robô parado.

Nessa seção serão descritos os detalhes do sistema desenvolvido, como a técnica utilizada e a visão geral do sistema, além dos resultados alcançados pelo sistema de *ground truth* desenvolvido a fim de mostrar sua eficiência em comparação com a posição real do robô, medida utilizando uma trena, e com o sistema de câmeras da categoria SSL.

#### 6.1.1 Técnica utilizada e visão geral do sistema

A Figura 26 mostra a arquitetura do software do sistema proposto, que é baseado num pacote do ROS2 que aplica um *scan matching* para encontrar a posição relativa de ambos os lasers. Em seguida, os dados de ambos os LiDARs são combinados em uma única imagem, em que diferentes operações morfológicas são aplicadas para melhorar a qualidade dela. Na sequência a etapa de detecção acontece para reconhecimento do padrão circular na imagem e, por fim, realiza-se o cálculo da coordenada global do robô, já que o sistema em um primeiro momento calcula a posição em relação ao posicionamento dos lasers.

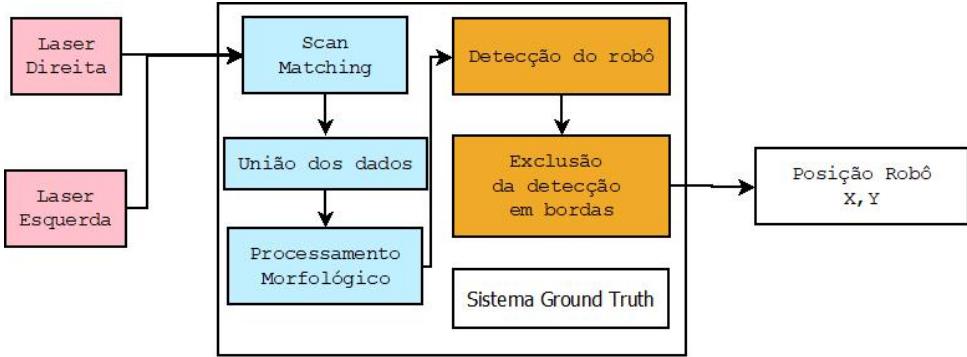


Figura 26 – Arquitetura do software.

O primeiro estágio do sistema é a utilização do *scan matching* para encontrar a transformação (translação, rotação e escala) entre as nuvens de pontos criadas a partir das leituras dos lasers que melhor encaixa os dados de ambos (NIETO; BAILEY; NEBOT, 2007). Essa etapa é decisiva no sistema visto que qualquer centímetro de diferença entre ambos os dados dos lasers pode levar a uma maior dificuldade para encontrar o padrão circular do robô ou pode adicionar um desvio na posição detectada.

O *scan matching* funciona a partir de duas nuvens de pontos, a fonte P e a alvo S. O objetivo do algoritmo é encontrar a transformação  $T = (R, t)$ , composta pela rotação R e translação t, que minimiza a diferença entre ambas as nuvens de pontos P e S, como descrito na Equação (31), em que  $r_i$  é o ponto mais perto na nuvem de ponto alvo S.

$$T^* = \arg \min \sum_{i=1}^n \|(Rp_i + t) - r_i\|^2 \quad (31)$$

O estágio seguinte do software é a união dos dados de ambos os lasers em uma imagem, chamada de imagem de pontos do laser, aplicando a transformação encontrada na etapa de *scan matching*. Para a obtenção dos dados dos sensores, foi utilizado o pacote do ROS2 desenvolvido pela SLAMTEC para ambos os lasers utilizados no sistema, já que é uma maneira mais fácil para comunicar com os sensores, inicializá-los, e receber os dados deles. O nó do ROS2 publica a mensagem do tipo *laser scan* que contém as distâncias de detecção pelo sensor e seus respectivos ângulos. A partir desses dados, é possível construir a imagem de pontos do laser seguindo a Equação (32), que demonstra como determinar as coordenadas de cada objeto detectado naquele *scan*, no caso  $r$  é a distância em metros e  $\theta$  é o ângulo respectivo à distância detectada em radianos.

$$\begin{cases} x = r * \cos(\theta) \\ y = r * \sin(\theta) \end{cases} \quad (32)$$

Depois de unir dos dados dos sensores, há uma camada para processar a imagem de pontos dos lasers antes do reconhecimento do formato circular no próximo estágio. O processamento morfológico de uma imagem é essencial para refinar a qualidade de imagens, redução de ruídos contidos nelas, e melhorar a análise estrutural delas, o que leva a uma extração de propriedades mais precisa em imagens (LOTUFO et al., 2023). Primeiro, o algoritmo aplica uma leve dilatação dos pontos a fim de conectar pontos próximos, em seguida há uma etapa de erosão para uma melhor detecção de formatos na imagem. Finalmente, um borrão delicado é aplicado na imagem de pontos dos lasers para suavização, reduzindo o ruído na imagem e a detecção de falsos círculos, assim como ajudando na detecção do padrão.

A próxima etapa do software é a detecção do robô. Esse estágio realiza a detecção do padrão circular, ou semi-circular, na imagem de pontos dos lasers. O método utilizado para detecção do padrão é a transformada de Hough (HT). A técnica, em um primeiro momento, foi criada para detecção de linhas em imagens binárias, mas outras variações foram sugeridas para detectar diferentes formatos, como os circulares (MUKHOPADHYAY; CHAUDHURI, 2015).

A HT para círculos utiliza da equação do círculo demonstrada na Equação (33), que o sistema calcula para cada ponto de borda na imagem pré-processada de acordo com os limites máximo e mínimo de raio definidos. Cada ponto de borda contribui com votos para possíveis círculos que ele pode fazer parte. Os votos são acumulados e os máximos locais representam as localizações dos centros e os raios dos círculos nas imagens (HASSANEIN et al., 2015).

$$(x - a)^2 + (y - b)^2 = r^2 \quad (33)$$

Na Figura 27 é possível observar dois momentos da arquitetura. A Figura 27a mostra os pontos dos lasers após os estágios de *scan matching* e união dos dados, mas antes da etapa de processamento morfológico. No caso, os pontos azuis são do laser à direita do campo, enquanto os pontos verdes são do laser à esquerda. Já a Figura 27b mostra a etapa de detecção, em que a HT é aplicada na imagem de pontos dos lasers para detecção do padrão circular.

A última etapa de pós-processamento é realizada para detectar e excluir falsos positivos nas bordas do campo, já que a maior parte dos pontos detectados pelo laser são os pontos da borda do campo. Basicamente, as coordenadas de fronteiras do campo são excluídos e a detecção é focada dentro do campo.

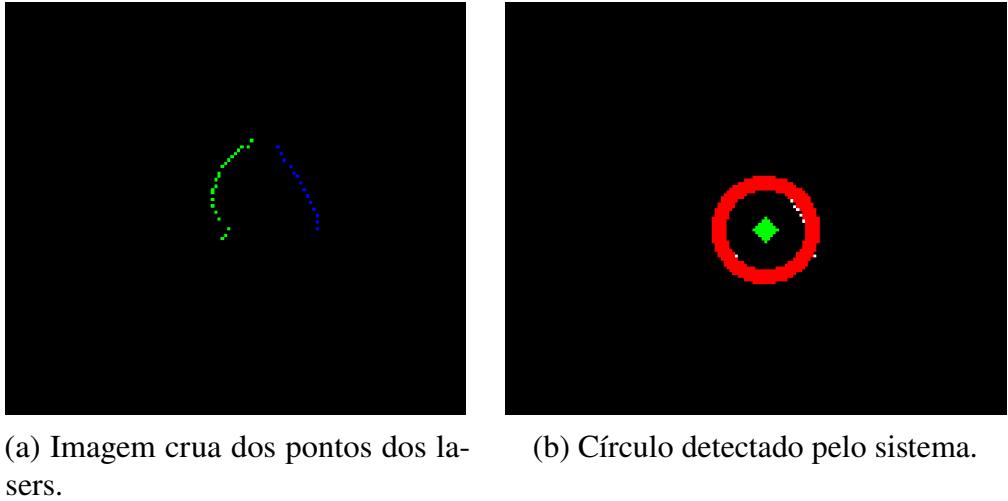


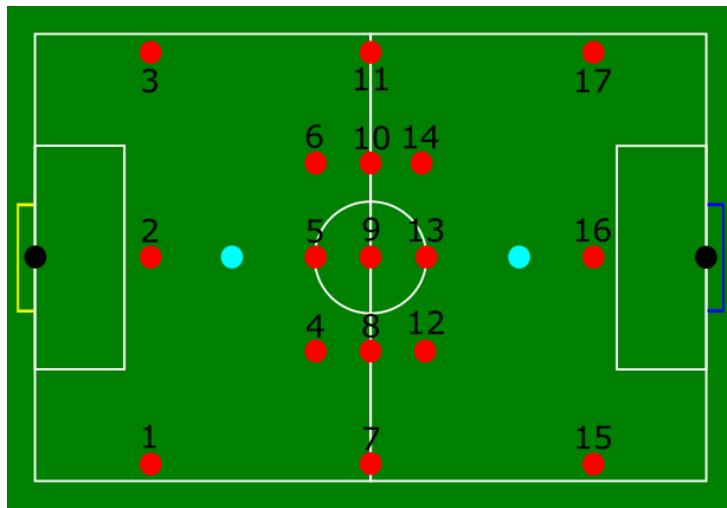
Figura 27 – Duas etapas da arquitetura desenvolvida.

### 6.1.2 Resultados do sistema desenvolvido

Para realizar a comparação, foram decididas 17 posições arbitrárias em campo, que estão demonstradas na Figura 28. Esses cenários foram escolhidos visando observar o comportamento do sistema de lasers desenvolvido em diferentes situações, no caso em posições mais centralizadas no campo (cenários centralizados, são as posições 4-6, 8-10 e 12-14), que são cenários praticamente equidistantes de ambos os lasers, e também em posições mais dispersas no campo (cenários dispersos, são as posições 1-3, 7, 11 e 15-17), onde os pontos estão consideravelmente mais perto de um laser do que outro, mas também nas beiradas do campo. Além disso, os cenários dispersos serviram para comparação com o sistema de câmeras da SSL, visto que por conta da distorção na utilização de lentes olho de peixe, há uma distorção maior nos cantos do campo.

Além dessa separação dos cenários, eles também foram separados em 3 conjuntos, que são chamados de quadrantes, que visam observar as diferenças de captura de ambos os sistemas na região de sobreposição das câmeras, localizada ao longo da região central do campo. O primeiro quadrante (posições 1-6) contém os cenários que somente a câmera da esquerda consegue capturar. O segundo quadrante (posições 7-14) é o conjunto dos cenários posicionados onde ambas câmeras conseguem capturar. Por fim, o terceiro quadrante (posições 15-17) representa os cenários em que somente a câmera à direita pode capturar. Essa separação também ajuda a entender como o sistema câmeras do SSL lida com a distorção separadamente. Espera-se que as coordenadas do SSL-Vision tenham um erro substancial com o distanciamento do centro das câmeras.

Figura 28 – Cenários de testes do sistema de ground truth.



Fonte: o Autor

Legenda: os pontos vermelhos numerados são os cenários utilizados para validar o sistema, os pontos pretos são as posições dos LiDARs em campo, enquanto os pontos azuis representam a projeção da posição das câmeras no campo.

Para poder comparar os sistemas com a posição real medida por uma trena, em cada posição foram capturadas 200 amostras do sistema de lasers, enquanto a saída do SSL-Vision também é adquirida. A média das amostras de ambos sistemas foi calculada para cada posição. Assim, é possível calcular o erro da média dos sistemas em relação ao que foi medido com a trena, assim como o desvio padrão para cada cenário a fim de analisar a estabilidade dos sistemas.

As Tabelas 8 e 9 resumem os resultados alcançados pelo SSL-Vision, sistema de lasers e a medição real para os eixos X e Y, respectivamente, nos 17 cenários de teste. A coluna Real indica as coordenadas reais do robô com o auxílio de uma trena, enquanto as colunas Laser e Visão mostram a média e o desvio padrão das amostras medidas pelos sistemas de lasers e câmeras, respectivamente.

A Tabela 10 mostra o erro e desvio padrão médios em todos os cenários, mas também separadamente para os conjuntos centralizado e disperso, nos eixos X e Y. Todas as medidas estão em centímetros. É possível analisar que o erro médio para o eixo X em ambos sistemas são comparáveis, com um diferença de aproximadamente 0.1cm entre ambos. Já para o eixo Y é possível observar uma diferença considerável entre eles, perto de 0.43cm. Essa diferença pode ser explicada pela distorção ocorrida nas bordas da imagem da câmera, o que aumenta o erro nesses pontos, já que eles possuem uma distância maior das câmeras no eixo Y do que no X.

Tabela 8 – Resultados alcançados para o eixo X no campo de teste.

Cenário	Real [cm]	Laser [cm]	Visão [cm]
1	-91,5	-91.797 +- 0.584	-92.945 +- 0.026
2	-88,1	-87.594 +- 0.437	-89.099 +- 0.023
3	-93	-92.510 +- 0.391	-93.397 +- 0.049
4	-55	-54.584 +- 0.703	-55.389 +- 0.037
5	-51	-50.321 +- 0.655	-51.323 +- 0.017
6	-54,5	-54.504 +- 0.926	-55.363 +- 0.037
7	0	1.092 +- 0.674	0.488 +- 0.012
8	0	0.390 +- 0.624	0.370 +- 0.073
9	0	0.552 +- 0.513	-0.158 +- 0.061
10	0	0.786 +- 0.478	-0.505 +- 0.036
11	0	0.972 +- 0.953	0.178 +- 0.025
12	61,3	61.859 +- 0.649	61.588 +- 0.017
13	52	51.964 +- 0.519	52.379 +- 0.011
14	53	53.742 +- 1.036	52.868 +- 0.046
15	101	101.653 +- 1.450	100.866 +- 0.015
16	99	99.683 +- 0.554	98.785 +- 0.017
17	97	97.526 +- 1.509	96.679 +- 0.018

Tabela 9 – Resultados alcançados para o eixo Y no campo de teste.

Cenário	Real [cm]	Laser [cm]	Visão [cm]
1	-138,5	-137.784 +- 0.726	-139.464 +- 0.047
2	0	0.844 +- 1.475	0.365 +- 0.018
3	132	131.688 +- 0.625	134.255 +- 0.022
4	-85	-84.104 +- 0.851	-85.172 +- 0.016
5	0	-0.518 +- 1.424	0.690 +- 0.036
6	81,5	80.801 +- 1.435	83.025 +- 0.018
7	-141	-140.488 +- 0.742	-139.829 +- 0.014
8	-82,5	-82.172 +- 0.978	-82.306 +- 0.061
9	0	0.668 +- 1.052	0.963 +- 0.041
10	83,5	84.431 +- 1.781	85.154 +- 0.028
11	125,5	125.017 +- 0.925	126.895 +- 0.083
12	-92,3	-92.848 +- 0.900	-91.665 +- 0.015
13	0	-0.568 +- 0.879	0.810 +- 0.014
14	85,3	85.046 +- 1.317	86.171 +- 0.053
15	-135,5	-135.906 +- 0.813	-135.158 +- 0.022
16	0	0.048 +- 0.598	1.236 +- 0.016
17	123,5	123.125 +- 1.015	124.690 +- 0.023

Os erros entre os conjuntos de cenários centralizado e disperso mostram que a câmera lida melhor com posições no centro do campo, o que é explicado pela distorção nas bordas das imagens das câmeras. Esse efeito não pode ser percebido analisando o sistema com lasers, mostrando que a detecção por eles é regular em todas as partes do campo, e prova que há

Tabela 10 – Erro e desvio padrão médios nos cenários.

	X - Laser	X - Visão	Y - Laser	Y - Visão
Erro médio	0,5519	0,4461	0,5356	0,9665
Erro - Centralizado	0,4626	0,3785	0,6011	0,8348
Erro - Disperso	0,6523	0,5221	0,4620	1,1147
Des. Padrão médio	0,7444	0,0305	1,0315	0,0310
Des. Padrão - Centralizado	0,6781	0,0372	1,1796	0,0313
Des. Padrão - Disperso	0,8190	0,0231	0,8648	0,0306

vantagem na utilização de dois lasers para melhorar a detecção do padrão circular do robô em campo.

Ao analisar o desvio padrão na Tabela 10, é possível notar que o sistema de câmeras da SSL apresenta resultados bem mais expressivos do que o sistema com LiDARs visto que a detecção foi aproximadamente 0.3mm dispersa da média analisando todos os cenários no eixo X, e menos que 0.4mm e 0.3mm nos conjuntos centralizado e disperso, respectivamente. Já para o eixo Y os resultados foram praticamente os mesmos, comparando com o sistema do laser, este foi pior também, mas os resultados continuam aceitáveis.

Quanto essa diferença exacerbada no desvio padrão dos dados do sistema do laser, é possível justificar isso pela resolução da imagem formada pelos dados dos lasers. Essa imagem formada é de 1000x1000 pixels, e cada pixel representa 0.533cm no campo por conta da escala utilizada para alocar os pontos na imagem. Logo, a menor diferença na detecção leva a aproximadamente 0.5cm na posição detectada do robô.

Analizando os quadrantes, as Tabelas 11 e 12 mostram os resultados resumidos para eles nos eixos X e Y, respectivamente. Notavelmente o quadrante pertencente à câmera esquerda é pior para o eixo X ao analisar o erro médio, sendo mais que o dobro de erro comparado com os quadrantes 2 e 3. A utilização de lente olho de peixe pode explicar esse efeito já que leva a uma distorção comparável à câmera direita. Para o eixo Y em todos os quadrantes, é possível observar que o erro médio é pior quando comparado com o sistema de lasers em pelo menos 50%. As medidas nas Tabelas 11 e 12 também estão em centímetros.

Tabela 11 – Resultados dos quadrantes para o eixo X.

	Erro X - Laser	Erro X - Vis.	Des. Padrão X - Laser	Des. Padrão X - Vis.
Quad. 1	0,3986	0,7360	0,6224	0,0315
Quad. 2	0,6411	0,3122	0,6807	0,0351
Quad. 3	0,6206	0,2233	1,1710	0,01667

Tabela 12 – Resultados dos quadrantes para o eixo Y.

	<b>Erro Y - Laser</b>	<b>Erro Y - Vis.</b>	<b>Des. Padrão Y - Laser</b>	<b>Des. Padrão Y - Vis.</b>
<b>Quad. 1</b>	0,6641	0,9951	1,0893	0,0261
<b>Quad. 2</b>	0,5844	1,0754	1,0956	0,0454
<b>Quad. 3</b>	0,3665	0,8473	0,9203	0,0238

Ao analisar o desvio padrão, não é notável qualquer influência do posicionamento das câmeras em ambos os eixos. Além disso, o SSL-Vision continua melhor que o sistema de lasers em todos os quadrantes para X e Y. Quando analisado para ambos os eixos, não é possível analisar diferenças significativas entre os quadrantes para o sistema de lasers, mostrando que a detecção realizada por ele é regular, e os erros acumulados em longas distâncias para um laser são compensados pelas leituras do outro laser.

## 6.2 IMPLEMENTAÇÃO DOS CENÁRIOS

Nesta Seção serão abordadas as implementações dos cenários de fusão de sensores propostos na Seção 5.2. Os estados definidos para os cenários desse trabalho seguem o que é descrito na Equação (34), em que  $\mathbf{X}$  é o vetor de estados do sistema no instante de tempo  $k$ . No caso, os estados do sistema são as posições globais  $x$  e  $y$  e orientação  $\theta$  do robô em campo, e pelas velocidades locais do robô  $v_x$  e  $v_y$ .

$$\mathbf{X}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \\ v_{x,k} \\ v_{y,k} \end{bmatrix} \quad (34)$$

### 6.2.1 Cenário 1 - Modelo + Visão

O primeiro cenário utiliza do modelo cinemático de um robô omnidirecional na fase de predição e os dados do sistema global de visão na fase de correção.

As entradas do sistema para esse cenário são as velocidades locais do robô nos eixos X e Y e a angular ( $v_{x,\text{teór}}$ ,  $v_{y,\text{teór}}$  e  $\omega_{\text{teór}}$ , respectivamente) calculadas pelo sistema de planejamento de trajetória da equipe, representada pelo vetor  $\mathbf{u}_k$ , conforme a Equação (35). O índice  $\text{teór}$  representa que as variáveis são adquiridas a partir do modelo cinemático do robô.

$$\mathbf{u}_k = \begin{bmatrix} v_{x,teór} \\ v_{y,teór} \\ \omega_{teór} \end{bmatrix} \quad (35)$$

A Equação (36) mostra a atualização dos estados na fase de predição do sistema de localização nesse cenário. Os índices  $k$  e  $k - 1$  representam o instante de tempo daquela variável, no caso os instantes atual e passado, respectivamente, tanto para esse quanto para os próximos cenários. Já o índice  $G$  representa as variáveis descritas nos eixos globais do campo, como no caso das velocidades nos eixos X e Y do campo  $V_{x,G}$  e  $V_{y,G}$ , respectivamente. A variável  $\Delta_t$  representa a variação de tempo, em segundos.

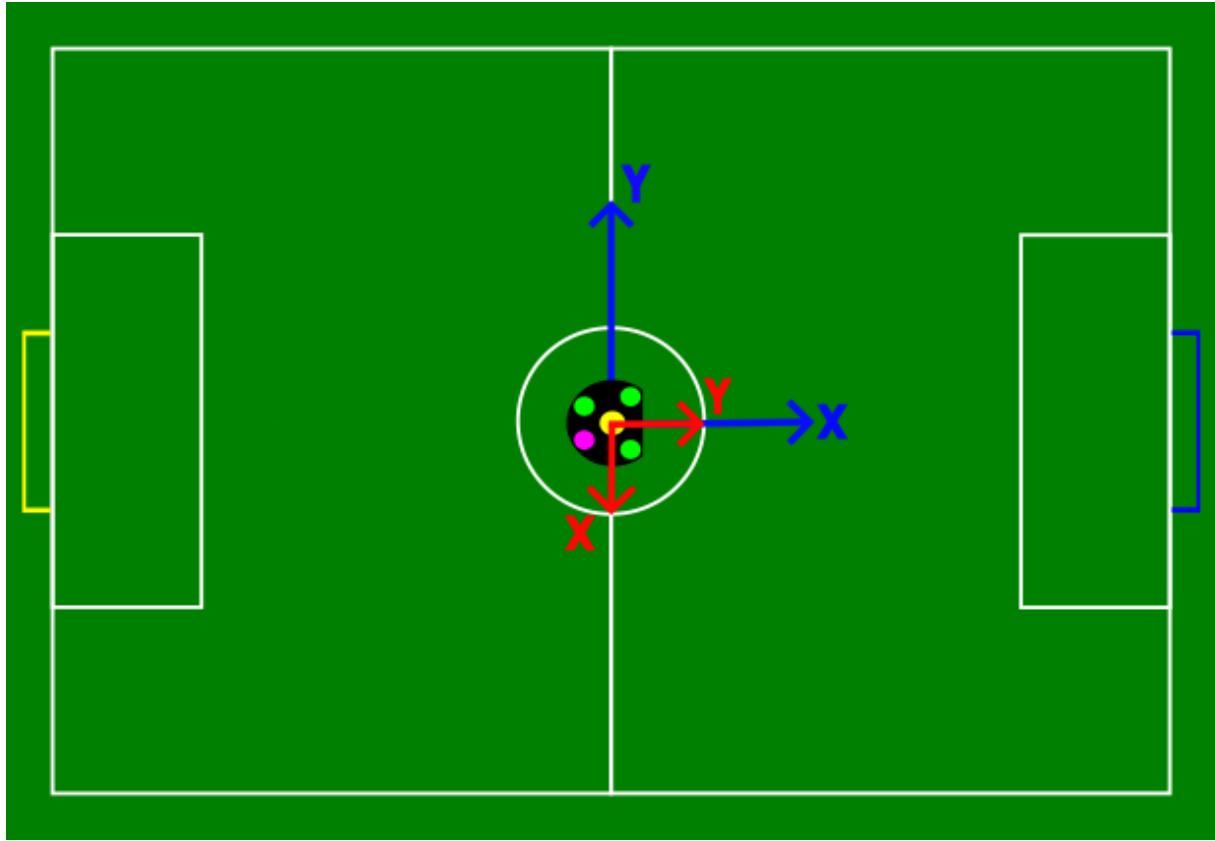
$$\begin{cases} x_k = x_{k-1} + V_{x,G}\Delta_t \\ y_k = y_{k-1} + V_{y,G}\Delta_t \\ \theta_k = \theta_{k-1} + \omega_{teór,k}\Delta_t \\ v_{x,k} = v_{x,teór,k} \\ v_{y,k} = v_{y,teór,k} \end{cases} \quad (36)$$

Para o cálculo das posições globais do robô é necessário que as velocidades estejam descritas em relação ao eixo do campo, no caso descritas como  $V_{x,G}$  e  $V_{y,G}$ , e são calculadas conforme a Equação (37). É importante notar a transformação feita subtraindo  $\frac{\pi}{2}$  do ângulo calculado do robô para alinhar os eixos locais do robô com os eixos globais do campo, conforme a Figura 29, em que o eixo em azul representa a convenção adotada pelo sistema de visão da SSL, enquanto o eixo em vermelho representa a convenção adotada pela equipe para o robô. Esse conceito de transformação será utilizado tanto nesse quanto nos próximos cenários.

$$\begin{cases} V_{x,G} = \cos(\theta_{k-1} - \frac{\pi}{2}) \times v_{x,teór,k-1} - \sin(\theta_{k-1} - \frac{\pi}{2}) \times v_{y,teór,k-1} \\ V_{y,G} = \sin(\theta_{k-1} - \frac{\pi}{2}) \times v_{x,teór,k-1} + \cos(\theta_{k-1} - \frac{\pi}{2}) \times v_{y,teór,k-1} \end{cases} \quad (37)$$

A partir disso, a matriz jacobiana  $G_k$  dos estados previstos em relação ao estado anterior segue a Equação (38).

Figura 29 – Diferença entre os eixos do campo e do robô.



Fonte: o Autor

$$G_k = \begin{bmatrix} 1 & 0 & (\cos(\theta)v_x\Delta_t - \sin(\theta)v_y\Delta_t) & \sin(\theta)\Delta_t & \cos(\theta)\Delta_t \\ 0 & 1 & (\sin(\theta)v_x\Delta_t - \cos(\theta)v_y\Delta_t) & -\cos(\theta)\Delta_t & \sin(\theta)\Delta_t \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (38)$$

Já a matriz de covariância dos ruídos do processo  $Q_k$  é definida na Equação (39).

$$Q_k = \begin{bmatrix} 2.34 \cdot 10^{-4} & 0 & 0 & 0 & 0 \\ 0 & 2.34 \cdot 10^{-4} & 0 & 0 & 0 \\ 0 & 0 & 2.23 \cdot 10^{-5} & 0 & 0 \\ 0 & 0 & 0 & 1 \cdot 10^{-4} & 0 \\ 0 & 0 & 0 & 0 & 1 \cdot 10^{-4} \end{bmatrix} \quad (39)$$

Quanto à fase de correção, nesse cenário são utilizados os dados de posição do sistema global de câmeras da SSL. Conforme a Equação (40), o sistema de visão providencia observações diretas tanto das posições globais  $x$  e  $y$  dos robôs, quanto da orientação  $\theta$ , com um certo ruído. O vetor de medições é apresentado pelo vetor  $\mathbf{z}_k$ .

$$\mathbf{z}_k = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \quad (40)$$

A partir disso, a matriz de transformação do estado real no estado observado  $C_k$  é definida na Equação (41). Já a matriz de covariância  $R_k$  das medições do sistema global de câmeras é definida na Equação (42).

$$C_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (41)$$

$$R_k = \begin{bmatrix} 1.35 \cdot 10^{-6} & 0 & 0 \\ 0 & 1.35 \cdot 10^{-6} & 0 \\ 0 & 0 & 9.74 \cdot 10^{-5} \end{bmatrix} \quad (42)$$

### 6.2.2 Cenário 2 - Modelo + Encoders

O segundo cenário também utiliza do modelo cinemático de um robô omnidirecional na fase de predição do EKF, mas utiliza os dados dos encoders acoplados às rodas na fase de correção.

A atualização dos estados na fase de predição do sistema de localização segue a Equação (36) definida na Seção 6.2.1, sendo as entradas definidas na Equação (35). A matriz jacobiana  $G_k$  é definida na Equação (38), enquanto a matriz dos ruídos do processo  $Q_k$  segue a Equação (39).

Para a fase de correção, são utilizados os dados dos encoders acoplados às rodas a fim de obter uma estimativa melhor das velocidades locais X e Y do robô e, assim das posições globais e da orientação, como demonstrado pelo vetor de medições  $\mathbf{z}_k$  na Equação (43). O índice  $enc$  nas variáveis demonstra que os estados do sistema são calculados a partir dos dados dos encoders, no caso as posições globais  $x_{enc}$  e  $y_{enc}$ , orientação  $\theta_{enc}$  e velocidades locais  $v_{x,enc}$  e  $v_{y,enc}$

$$\mathbf{z}_k = \begin{bmatrix} x_{enc,k} \\ y_{enc,k} \\ \theta_{enc,k} \\ v_{x,enc,k} \\ v_{y,enc,k} \end{bmatrix} \quad (43)$$

A Equação (44) demonstra como os estados para a fase de correção são atualizados com as leituras dos encoders, sendo que  $\Delta_t$  representa o intervalo de tempo entre uma etapa de correção e outra. Já a variável  $\omega_{enc}$  representa a velocidade angular provida pelos encoders.

$$\left\{ \begin{array}{l} x_{enc,k} = x_{enc,k-1} + V_{x,enc,G}\Delta_t \\ y_{enc,k} = y_{enc,k-1} + V_{y,enc,G}\Delta_t \\ \theta_{enc,k} = \theta_{enc,k-1} + \omega_{enc,k}\Delta_t \\ v_{x,enc,k} = v_{x,enc,k} \\ v_{y,enc,k} = v_{y,enc,k} \end{array} \right. \quad (44)$$

O cálculo das velocidades globais para determinar as posições globais segue a ideia da Equação (37). Porém, na Equação (45), para o cálculo de  $V_{x,enc,G}$  e  $V_{y,enc,G}$ , as velocidades locais utilizadas são providas a partir dos encoders.

$$\left\{ \begin{array}{l} V_{x,enc,G} = \cos(\theta_{enc,k-1} - \frac{\pi}{2}) \times v_{x,enc,k} - \sin(\theta_{enc,k} - \frac{\pi}{2}) \times v_{y,enc,k} \\ V_{y,enc,G} = \sin(\theta_{enc,k-1} - \frac{\pi}{2}) \times v_{x,enc,k} + \cos(\theta_{enc,k} - \frac{\pi}{2}) \times v_{y,enc,k} \end{array} \right. \quad (45)$$

A partir disso, a matriz  $C_k$  de transformação do estado real no estado observado segue a Equação (46).

$$C_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (46)$$

A matriz de covariância dos ruídos das medições  $R_k$  desse cenário é definida na Equação (47).

$$R_k = \begin{bmatrix} 2.50 \cdot 10^{-6} & 0 & 0 & 0 & 0 \\ 0 & 2.50 \cdot 10^{-6} & 0 & 0 & 0 \\ 0 & 0 & 5 \cdot 10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 2.50 \cdot 10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 2.50 \cdot 10^{-6} \end{bmatrix} \quad (47)$$

### 6.2.3 Cenário 3 - Modelo + IMU

O terceiro cenário também utiliza do modelo cinemático de um robô omnidirecional na fase de predição do EKF, enquanto na fase de correção são utilizados os dados da IMU presente na placa principal do robô.

Para a fase de predição, o conceito segue o que foi descrito nas Seções 6.2.1 e 6.2.2. Já para a fase de correção, são utilizadas as leituras da IMU (giroscópio + acelerômetro) para obter uma estimativa dos estados do sistema. As variáveis com índice  $imu$  são as calculadas a partir dos dados da IMU, no caso as posições globais  $x_{imu}$  e  $y_{imu}$ , orientação  $\theta_{imu}$  e velocidades locais  $v_{x,imu}$  e  $v_{y,imu}$  do robô, conforme o que é descrito na Equação (48) no vetor  $\mathbf{z}_k$ .

$$\mathbf{z}_k = \begin{bmatrix} x_{imu,k} \\ y_{imu,k} \\ \theta_{imu,k} \\ v_{x,imu,k} \\ v_{y,imu,k} \end{bmatrix} \quad (48)$$

A Equação (49) demonstra como os estados para a fase de correção são atualizados com as leituras do giroscópio e acelerômetro. As variáveis  $V_{x,imu,G}$  e  $V_{y,imu,G}$  são as velocidades globais calculadas a partir dos dados da IMU, enquanto as variáveis  $A_{x,imu,G}$  e  $A_{y,imu,G}$  indicam as acelerações calculadas nos eixos globais do campo.  $\Delta_t$  representa o intervalo de tempo entre os instantes  $k$  e  $k - 1$ .

$$\begin{cases} x_{imu,k} = x_{imu,k-1} + V_{x,imu,G}\Delta_t + \frac{1}{2}A_{x,imu,G}\Delta_t^2 \\ y_{imu,k} = y_{imu,k-1} + V_{y,imu,G}\Delta_t + \frac{1}{2}A_{y,imu,G}\Delta_t^2 \\ \theta_{imu,k} = \theta_{imu,k-1} + \omega_{imu,k}\Delta_t \\ v_{x,imu,k} = v_{x,imu,k-1} + a_{x,k}\Delta_t \\ v_{y,imu,k} = v_{y,imu,k-1} + a_{y,k}\Delta_t \end{cases} \quad (49)$$

No caso da IMU, tanto as velocidades globais quanto as leituras de aceleração precisam ser transformadas para os eixos globais do campo. Sendo assim, as Equações (50) e (51) demonstram a transformação feita para ambos os componentes. As variáveis  $a_x$  e  $a_y$  representam as acelerações nos eixos locais X e Y do robô.

$$\begin{cases} V_{x,imu,G} = \cos(\theta_{imu,k-1} - \frac{\pi}{2}) \times v_{x,imu,k-1} - \sin(\theta_{imu,k-1} - \frac{\pi}{2}) \times v_{y,imu,k-1} \\ V_{y,imu,G} = \sin(\theta_{imu,k-1} - \frac{\pi}{2}) \times v_{x,imu,k-1} + \cos(\theta_{imu,k-1} - \frac{\pi}{2}) \times v_{y,imu,k-1} \end{cases} \quad (50)$$

$$\begin{cases} A_{x,imu,G} = \cos(\theta_{imu,k-1} - \frac{\pi}{2}) \times a_{x,k} - \sin(\theta_{imu,k-1} - \frac{\pi}{2}) \times a_{y,k} \\ A_{y,imu,G} = \sin(\theta_{imu,k-1} - \frac{\pi}{2}) \times a_{x,k} + \cos(\theta_{imu,k-1} - \frac{\pi}{2}) \times a_{y,k} \end{cases} \quad (51)$$

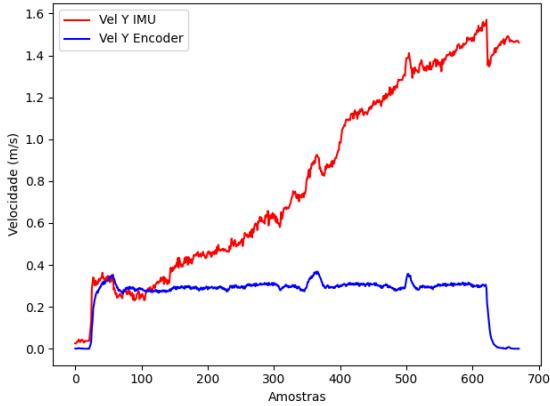
A partir disso, a matriz  $C_k$  de transformação do estado real no estado observado segue a Equação (52), já que a correção a partir dos dados da IMU provê observações diretas sobre todos os estados do sistema.

$$C_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (52)$$

A matriz de covariância dos ruídos das medições  $R_k$  desse cenário é definida na Equação (53).

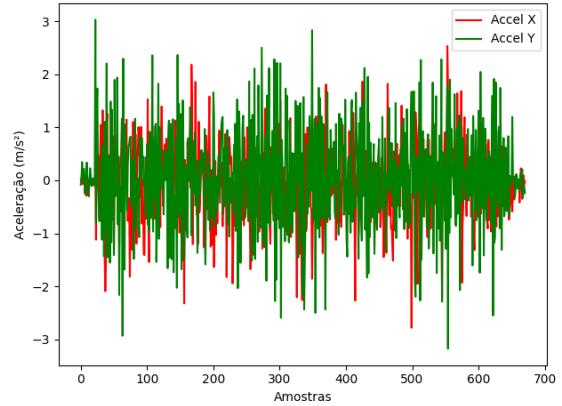
$$R_k = \begin{bmatrix} 7.04 \cdot 10^{-5} & 0 & 0 & 0 & 0 \\ 0 & 7.04 \cdot 10^{-5} & 0 & 0 & 0 \\ 0 & 0 & 4.04 \cdot 10^{-4} & 0 & 0 \\ 0 & 0 & 0 & 2.5 \cdot 10^{-3} & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-3} \end{bmatrix} \quad (53)$$

Figura 30 – Diferença das velocidades calculadas pelo acelerômetro e pelos encoders.



Fonte: o Autor

Figura 31 – Acelerações obtidas durante o teste.



Fonte: o Autor

É importante pontuar que ao utilizar a IMU para a fase de correção, tanto nesse quanto nos outros cenários, é feita a integração da aceleração provida pelo acelerômetro a fim de estimar a velocidade e posição do robô. Entretanto, ao fazer essa integração, por conta do alto ruído presente nos dados do sensor, acaba-se acumulando um grande erro em um curto espaço de tempo. Esse efeito pode ser observado na Figura 30, que compara a velocidade local no eixo Y obtida a partir tanto da IMU (vermelho) quanto dos encoders (azul), enquanto a Figura 31 mostra as leituras do acelerômetro durante o trajeto. É notável que durante o começo do movimento a integração da aceleração tem resultados parecidos com a velocidade dos encoders, mas depois de um tempo os erros da integração começam a crescer desenfreadamente.

Para impedir esse acúmulo de erro desenfreado a longo prazo, decidiu-se que as velocidades locais  $v_{x,imu}$  e  $v_{y,imu}$  calculadas a partir da IMU, a cada 5 ciclos de correção performados pelo EKF, são definidas como as velocidades locais  $v_x$  e  $v_y$  estimadas pelo filtro, seguindo a Equação (54).

$$\begin{cases} v_{x,imu,k} = v_{x,k} \\ v_{y,imu,k} = v_{y,k} \end{cases} \quad (54)$$

#### 6.2.4 Cenário 4 - IMU + Visão

Nesse cenário 4, os dados da IMU são utilizados na fase de predição do EKF, enquanto na fase de correção são utilizados os dados do sistema global de câmeras da categoria SSL.

Para a fase de predição do EKF, os dados do acelerômetro são utilizados para predizer tanto as posições globais  $x$  e  $y$  quanto as velocidades locais  $v_x$  e  $v_y$  nos X e Y do robô, enquanto os dados do giroscópio são utilizados para predizer a orientação  $\theta$  do robô em campo. A Equação (56) mostra como é feita a atualização de cada um dos estados com os dados da IMU, sabendo do intervalo de tempo  $\Delta_t$  entre um instante e outro. O vetor de entrada do sistema  $\mathbf{u}_k$  é composto pelas acelerações locais  $a_x$  e  $a_y$  nos eixos X e Y do robô, assim como a velocidade angular  $\omega$  do robô, como está descrito na Equação (55).

$$\mathbf{u}_k = \begin{bmatrix} a_x \\ a_y \\ \omega \end{bmatrix} \quad (55)$$

$$\begin{cases} x_k = x_{k-1} + V_{x,G}\Delta_t + \frac{1}{2}A_{x,imu,G}\Delta_t^2 \\ y_k = y_{k-1} + V_{y,G}\Delta_t + \frac{1}{2}A_{y,imu,G}\Delta_t^2 \\ \theta_k = \theta_{k-1} + \omega_k\Delta_t \\ v_{x,k} = v_{x,k-1} + a_{x,k}\Delta_t \\ v_{y,k} = v_{y,k-1} + a_{y,k}\Delta_t \end{cases} \quad (56)$$

O cálculo das velocidades globais desse cenário segue o que é descrito na Equação (37), enquanto a transformação das acelerações locais para globais é demonstrada na Equação (57).

$$\begin{cases} A_{x,imu,G} = \cos(\theta_{k-1} - \frac{\pi}{2}) \times a_{x,k} - \sin(\theta_{k-1} - \frac{\pi}{2}) \times a_{y,k} \\ A_{y,imu,G} = \sin(\theta_{k-1} - \frac{\pi}{2}) \times a_{x,k} + \cos(\theta_{k-1} - \frac{\pi}{2}) \times a_{y,k} \end{cases} \quad (57)$$

A partir disso, a matriz jacobiana  $G_k$  dos estados previstos em relação ao estado anterior segue a Equação (58).

$$G_k = \begin{bmatrix} 1 & 0 & \cos(\theta)\left(v_x\Delta t + a_x\frac{\Delta t^2}{2}\right) - \sin(\theta)\left(v_y\Delta t + a_y\frac{\Delta t^2}{2}\right) & \sin(\theta)\Delta t & \cos(\theta)\Delta t \\ 0 & 1 & \sin(\theta)\left(v_x\Delta t + a_x\frac{\Delta t^2}{2}\right) + \cos(\theta)\left(v_y\Delta t + a_y\frac{\Delta t^2}{2}\right) & -\cos(\theta)\Delta t & \sin(\theta)\Delta t \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (58)$$

A matriz das covariâncias dos ruídos  $Q_k$  do modelo é definida conforme a Equação (59).

$$Q_k = \begin{bmatrix} 8.39 \cdot 10^{-3} & 0 & 0 & 0 & 0 \\ 0 & 8.39 \cdot 10^{-3} & 0 & 0 & 0 \\ 0 & 0 & 2.01 \cdot 10^{-2} & 0 & 0 \\ 0 & 0 & 0 & 5 \cdot 10^{-2} & 0 \\ 0 & 0 & 0 & 0 & 5 \cdot 10^{-2} \end{bmatrix} \quad (59)$$

Quanto à fase de correção, nesse cenário são utilizados os dados de posição do sistema global de câmeras da SSL, conforme foi descrito na Seção 6.2.1 nas Equações (40) e (41), que descrevem o vetor de medições  $\mathbf{z}_k$  e a matriz de transformação do estado real no estado observado  $C_k$ , respectivamente. A matriz de covariâncias das medições no caso dos encoders é definida na Seção 6.2.1 na Equação (42).

### 6.2.5 Cenário 5 - IMU + Encoders

No quinto cenário, continua-se utilizando os dados da IMU (acelerômetro + giroscópio) para a fase de predição do Filtro de Kalman Estendido, enquanto os dados de velocidade dos encoders são utilizados na fase de correção do EKF.

A predição segue o que foi descrito na Seção 6.2.4, especialmente as Equações (55), que define o vetor de controle  $\mathbf{u}_k$ , (56), que define como os estados do sistema são atualizados na fase de predição, (57), que mostra como as acelerações locais são transformadas em globais para o cálculo das posições, e (58), que define a matriz das jacobianas dos estados em relação ao estado anterior.

Para fase de correção, os conceitos utilizados seguem o que foi descrito na Seção 6.2.2, especialmente as Equações (43), que define o vetor de medições do sistema, (44), que demonstra como as medições são atualizadas com as leituras dos encoders, e (46), que define a matriz de transformação do estado real no estado observado. A matriz de covariâncias das medições no caso dos encoders é definida na Seção 6.2.2 na Equação (47).

### 6.2.6 Cenário 6 - Encoders + Visão

No sexto cenário, os dados dos encoders acoplados às rodas do robô providenciam leituras das velocidades locais lineares e angular na fase de predição, enquanto na fase de correção são utilizados os dados do sistema global de câmeras da SSL.

Para a fase de predição do EKF, utilizam-se os dados provenientes dos encoders acoplados às rodas, transformando a leitura deles nas velocidades  $v_{x,enc}$  e  $v_{y,enc}$  descritas nos eixos do robô e angular  $\omega_{enc}$ , como descrito na Equação (60), que descreve o vetor das variáveis de controle do sistema nesse cenário. Os estados do sistema são atualizados conforme a Equação (61) sabendo do intervalo de tempo  $\Delta_t$  entre etapas de predição.

$$\mathbf{u}_k = \begin{bmatrix} v_{x,enc} \\ v_{y,enc} \\ \omega_{enc} \end{bmatrix} \quad (60)$$

$$\begin{cases} x_k = x_{k-1} + V_{x,G}\Delta_t \\ y_k = y_{k-1} + V_{y,G}\Delta_t \\ \theta_k = \theta_{k-1} + \omega_{enc,k}\Delta_t \\ v_{x,k} = v_{x,enc,k-1} \\ v_{y,k} = v_{y,enc,k-1} \end{cases} \quad (61)$$

O cálculo das velocidades globais desse cenário segue o que foi descrito na Equação (37).

A partir disso, a matriz jacobiana  $G_k$  dos estados previstos em relação ao estado anterior segue a Equação (62).

$$G_k = \begin{bmatrix} 1 & 0 & \cos(\theta)(v_x\Delta t) - \sin(\theta)(v_y\Delta t) & \sin(\theta)\Delta t & \cos(\theta)\Delta t \\ 0 & 1 & \sin(\theta)(v_x\Delta t) + \cos(\theta)(v_y\Delta t) & -\cos(\theta)\Delta t & \sin(\theta)\Delta t \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (62)$$

A matriz  $R_k$  das covariâncias dos ruídos do modelo é definida conforme a Equação (63).

$$R_k = \begin{bmatrix} 7.04 \cdot 10^{-5} & 0 & 0 & 0 & 0 \\ 0 & 7.04 \cdot 10^{-5} & 0 & 0 & 0 \\ 0 & 0 & 4.04 \cdot 10^{-4} & 0 & 0 \\ 0 & 0 & 0 & 2.5 \cdot 10^{-3} & 0 \\ 0 & 0 & 0 & 0 & 2.5 \cdot 10^{-3} \end{bmatrix} \quad (63)$$

Na fase de correção do EKF é utilizada a informação da posição global provida pelo sistema global de câmeras da SSL, conforme foi descrito na Seção 6.2.1, especialmente nas Equações (40) e (41), que descrevem o vetor de medições  $\mathbf{z}_k$  e a matriz de transformação do estado real no estado observado  $C_k$ , respectivamente. A matriz de covariâncias das medições no caso dos encoders é definida na Seção 6.2.1 na Equação (42).

### 6.2.7 Cenário 7 - Encoders + IMU

No sétimo, e último, cenário de fusão de sensores, os dados providos das leituras dos encoders ligados às rodas são utilizados na fase de predição do EKF, transformando as leituras das velocidades das rodas nas velocidades locais lineares e angular do robô. Já na fase de correção são utilizados dados do acelerômetro e giroscópio disponíveis no hardware do robô.

Quanto à fase de predição do EKF, segue o que foi descrito na Seção 6.2.6, especialmente nas Equações (60), que define o vetor de controle  $\mathbf{u}_k$ , (61), que define a atualização dos estados do sistema na fase de predição, e (62), que define a matriz das jacobianas dos estados em relação ao estado anterior.

Na fase de correção segue o que foi descrito na Seção 6.2.3, especialmente as Equações (48), que define o vetor de medições do sistema  $\mathbf{z}_k$ , (49), que define como as medições são atualizadas conforme as leituras da IMU, (50), que mostra o cálculo das velocidades globais com os dados da IMU, (51), que demonstra o cálculo das acelerações globais, e (52), que define a matriz de transformação do estado real no estado observado  $C_k$ . A matriz de covariâncias das medições no caso da IMU é definida na Seção 6.2.3 na Equação (53).

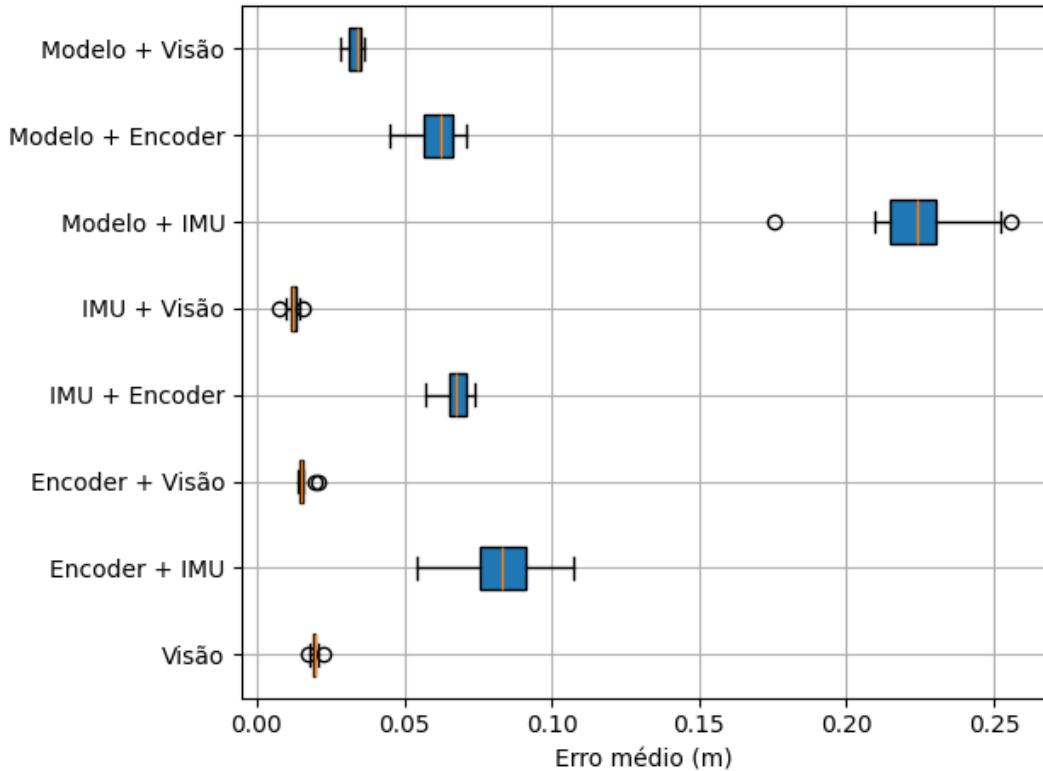
## 6.3 TESTE COM TRAJETO QUADRADO MENOR

Nesta seção será feita a análise do teste feito num trajeto quadrado menor. O intuito desse teste é analisar o desempenho das diferentes combinações de sensores propostas para localização do robô numa região do campo de sobreposição de duas câmeras, que acaba levando a duplicações nas leituras das posições dos robôs.

Na Figura 32 são mostrados os resultados obtidos no trajeto com quadrado menor em relação à distância média entre o que é calculado nas combinações propostas, o sistema atual da equipe e o sistema de *ground truth* desenvolvido. Nela, é possível notar que os cenários que utilizam sensores, como IMU e encoders, na fase de predição e o sistema global de câmeras da liga na fase de correção conseguem resultados mais satisfatórios, até em comparação com o que

é utilizado atualmente pela equipe somente com o sistema global de câmeras (Cenário 8). Esses resultados são notórios pelo baixo erro médio alcançado, assim como a uniformidade dos dados.

Figura 32 – Erro médio de distância no teste com trajeto do quadrado menor.



Fonte: o Autor

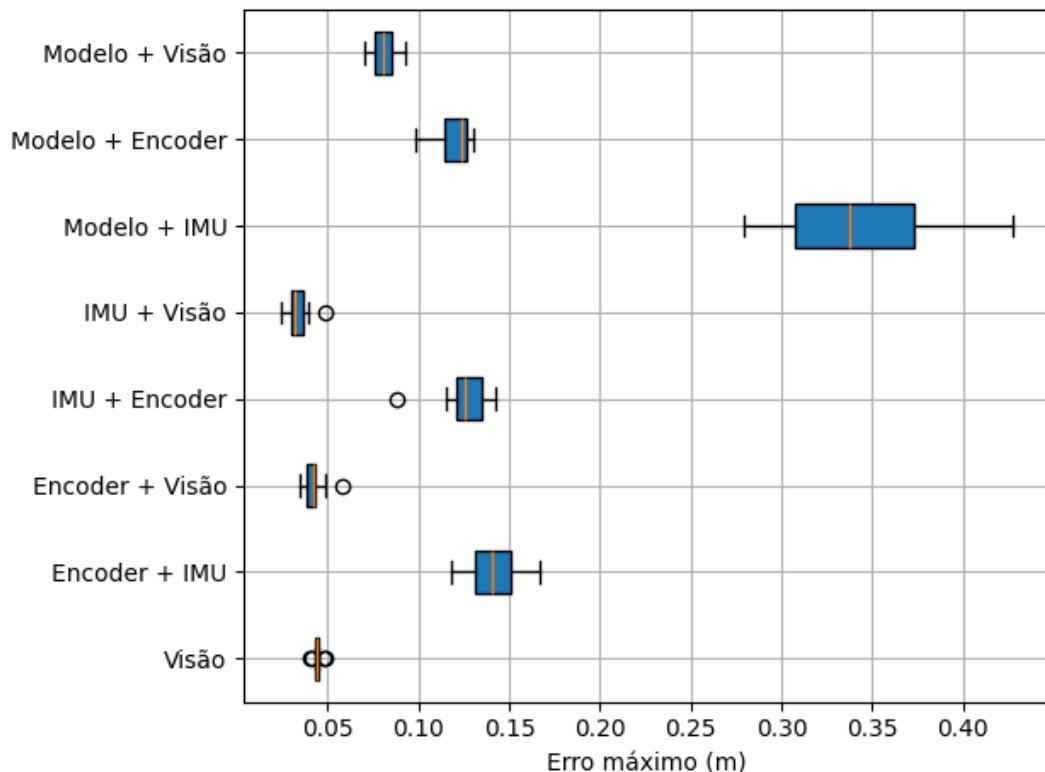
Além disso, é possível notar que a utilização de sensores na fase de predição do Filtro de Kalman Estendido traz melhores resultados que a utilização do modelo teórico cinemático do robô omnidirecional da categoria SSL, visto que as leituras de velocidade e aceleração adquiridas pelos sensores encoder e IMU provém estimativas melhores do que a utilização da velocidade teórica enviada ao robô. Esses resultados podem ser explicados pelas não-linearidades inseridas no movimento de um robô com rodas omnidirecionais, como o atrito, por exemplo.

Sobre os cenários que não utilizam do sistema global de câmeras da SSL, é possível notar que os erros médios desses cenários são maiores que 5cm, sendo o Cenário 3 (modelo + IMU) o pior deles, o que indica um problema na utilização do acelerômetro na fase de correção. Apesar disso, para escopos fora da SSL, os erros apresentados nos Cenários 2, 5 e 7 são erros considerados aceitáveis, mostrando que eles podem ser usados em cenários em que não é possível a utilização de um sensor que providencia leituras globais de posição, como câmeras e GPS.

Além disso, também a partir da Figura 32, é possível notar que os cenários que utilizam do sistema global de câmeras da SSL possuem menos variações em relação ao erro do que os cenários que utilizam apenas os sensores embarcados no robô. Isso demonstra que o erro médio ao longos dos testes nas combinações que utilizam a visão é mais estável do que nas outras combinações.

A Figura 33 mostra a distribuição do erro máximo em metros nos testes do trajeto do quadrado menor em cada combinação. O gráfico corrobora com o que foi discutido anteriormente sobre a utilização do sistema global de câmeras na fase de correção do EKF, em que os cenários que utilizam do SSL-Vision possuem os menores erros máximos ao longo dos trajetos. Além disso, também é possível notar que a utilização dos sensores na fase de predição é superior à utilização do modelo cinemático do robô omnidirecional.

Figura 33 – Erro máximo no teste com trajeto do quadrado menor.

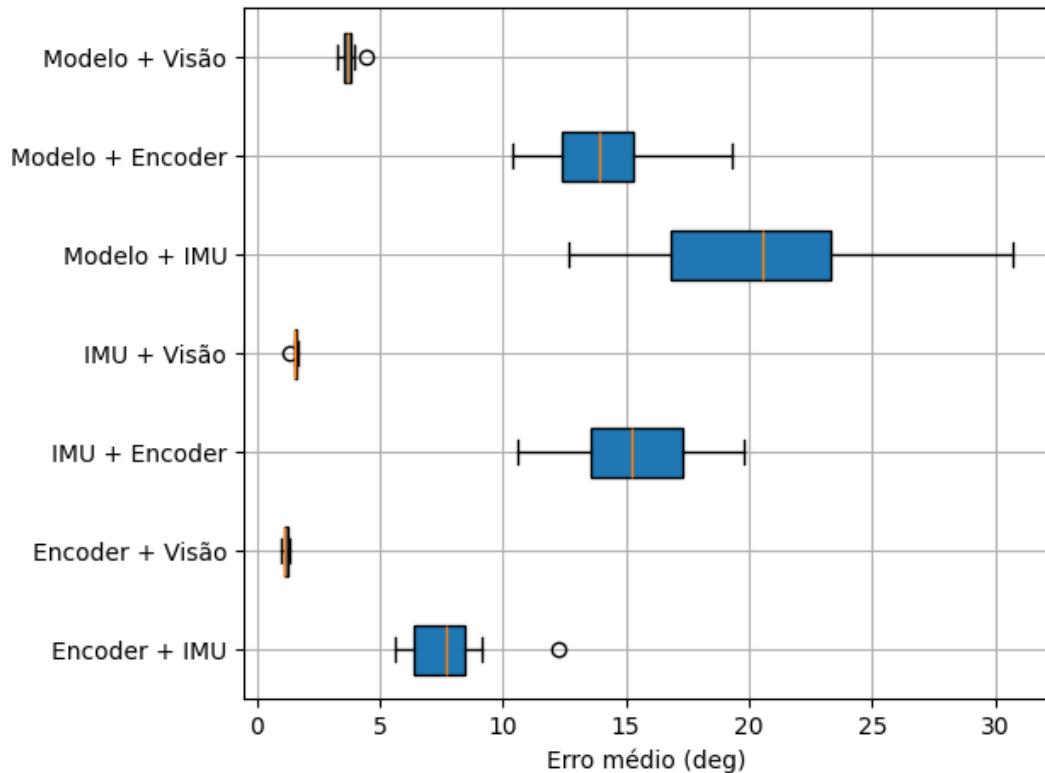


Fonte: o Autor

Apesar do sistema de *ground truth* não ser capaz de capturar a orientação do robô em campo, foi realizada uma comparação das combinações com o próprio sistema de visão com relação à orientação, apenas para título de comparação. Os resultados do erro médio da orientação

calculada pelo nos cenários do sistema de fusão de sensores e o que é lido pela visão estão descritos na Figura 34.

Figura 34 – Erro médio de orientação no teste com trajeto do quadrado menor.



Fonte: o Autor

Assim como na análise do erro médio de distância, é possível analisar que a utilização dos dados do sistema global de câmeras na fase de correção traz melhores resultados do que as combinações que fazem utilização apenas do modelo, encoders ou IMU. Nesses cenários sem o sistema de câmeras, é possível notar que o erro médio é maior do que 5° em todos eles, sendo que, fora o Cenário 7, em todos o erro médio é maior que 10°.

Por fim, a Tabela 13 mostra os resultados gerais obtidos na região de sobreposição de 2 câmeras com relação ao erro médio de distância, erro médio de orientação e o tempo de atualização médio. Quanto ao erro médio de distância é possível analisar que a utilização de sensores, como IMU e encoders (Cenários 4 e 6), consegue resultados levemente superiores comparados com os resultados atingidos pelo sistema atual utilizado pela equipe (Cenário 8), que utiliza somente os dados do sistema de câmeras da liga.

Tabela 13 – Resumo dos resultados obtidos dos cenários no teste do quadrado menor.

<b>Cenário</b>	Distância - Erro			Orientação - Erro		$T_{att}$ [ms]
	<b>Média [m]</b>	<b>Máximo [m]</b>	<b>Desv. Pad. [m]</b>	<b>Média [deg]</b>	<b>Máximo [deg]</b>	
Modelo + Visão	0,033067	0,080706	0,037178	3,725156	9,467714	3,45970
Modelo + Encoders	0,060239	0,119910	0,061663	13,88283	31,08587	3,58013
Modelo + IMU	0,223313	0,343750	0,199025	20,679713	57,796915	3,59125
IMU + Visão	0,012167	0,033837	0,014022	1,541681	4,788489	3,51205
IMU + Encoders	0,067586	0,125119	0,061642	15,263899	30,86708	3,59578
Encoders + Visão	0,015852	0,042467	0,018073	1,146865	4,148206	3,45462
Encoders + IMU	0,083122	0,140408	0,076724	7,710782	18,938697	3,59147
Visão	0,019585	0,044054	0,018842	-	-	16,2879

Além disso, quanto ao tempo de atualização, os resultados obtidos nos cenários com fusão de sensores conseguem um tempo praticamente 5x menor do que o sistema atual da equipe, visto que o EKF acontece de forma embarcada ao robô.

#### 6.4 TESTE COM TRAJETO QUADRADO MAIOR

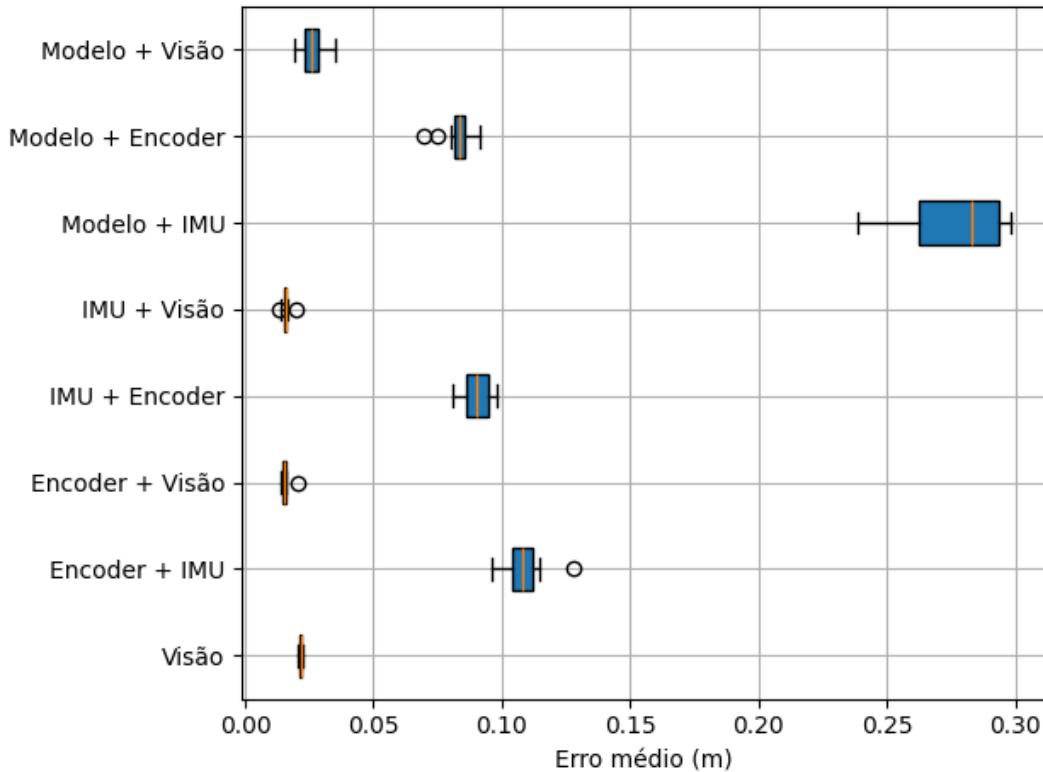
Nesta seção será feita a análise do teste do trajeto quadrado maior em campo. O intuito desse teste é analisar o desempenho das diferentes combinações de sensores propostas para localização do robô numa região do campo em que a distorção da imagem do sistema global de câmeras da SSL é maior, o que pode levar a maiores erros na detecção da posição do robô em campo.

Na Figura 35 é mostrada a distribuição do erro médio de distância nos testes com quadrado maior nos oitos cenários propostos neste projeto. É notável que a utilização dos dados do SSL-Vision leva a erros médios menores comparado com os cenários que não utilizam os dados do sistema de câmeras. Analisando os cenários que não utilizam o SSL-Vision, é possível notar erros médios maiores do que no teste da Seção 6.3, o que mostra que a utilização de um sensor que providencie leituras globais de posição evita erros acumulativos em relação à distância percorrida.

Assim como nos resultados apresentados quanto ao erro médio nos testes com quadrado menor na Seção 6.3, é possível analisar resultados levemente melhores ao utilizar sensores na fase de predição, como encoders e IMU, junto ao SSL-Vision quando comparado com o sistema atual da equipe que utiliza somente os dados do sistema de visão sem fusão.

A Figura 36 traz a distribuição dos erros máximos de distância nos cenários analisados. O gráfico corrobora com os resultados do gráfico de distribuição do erro médio de distância nos cenários analisados, é notável também que a utilização do SSL-Vision diminui consideravelmente

Figura 35 – Erro médio de distância no teste com trajeto do quadrado maior.

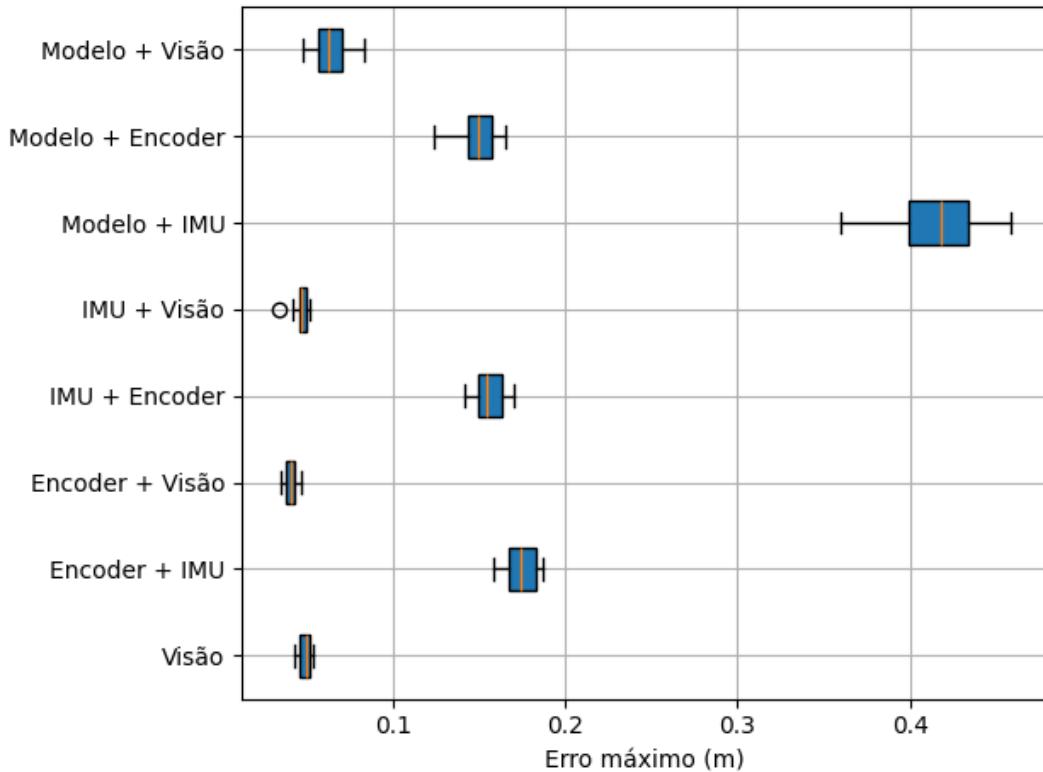


Fonte: o Autor

o erro máximo ao longo do trajeto. Tanto o Cenário 8, que é o sistema atual da equipe, como os cenários que utilizam o SSL-Vision na fase de correção da fusão de sensores, possuem erro máximo menor que 5cm, que representa menos que 1/3 do tamanho máximo de um robô típico da categoria SSL.

A Figura 37 traz a média do erro de orientação em relação ao sistema de visão a título de comparação quanto a orientação. Assim como na análise quanto ao erro de distância, a utilização do SSL-Vision na fase de correção traz resultados melhores quanto a uniformidade e média. Apesar disso, é possível notar que o Cenário 7 traz resultados aceitáveis quanto a orientação (menor que 10°), mostrando que a utilização do encoder na fase de predição é melhor do que na fase de correção, o que pode ser explicado pela integração direta da velocidade angular calculada a partir das leituras dos encoders nas rodas, diferente do que acontece na fase de predição, que a estimativa da orientação leva em conta o estado passado do EKF, o que aumenta o erro acumulado.

Figura 36 – Erro máximo no teste com trajeto do quadrado maior.



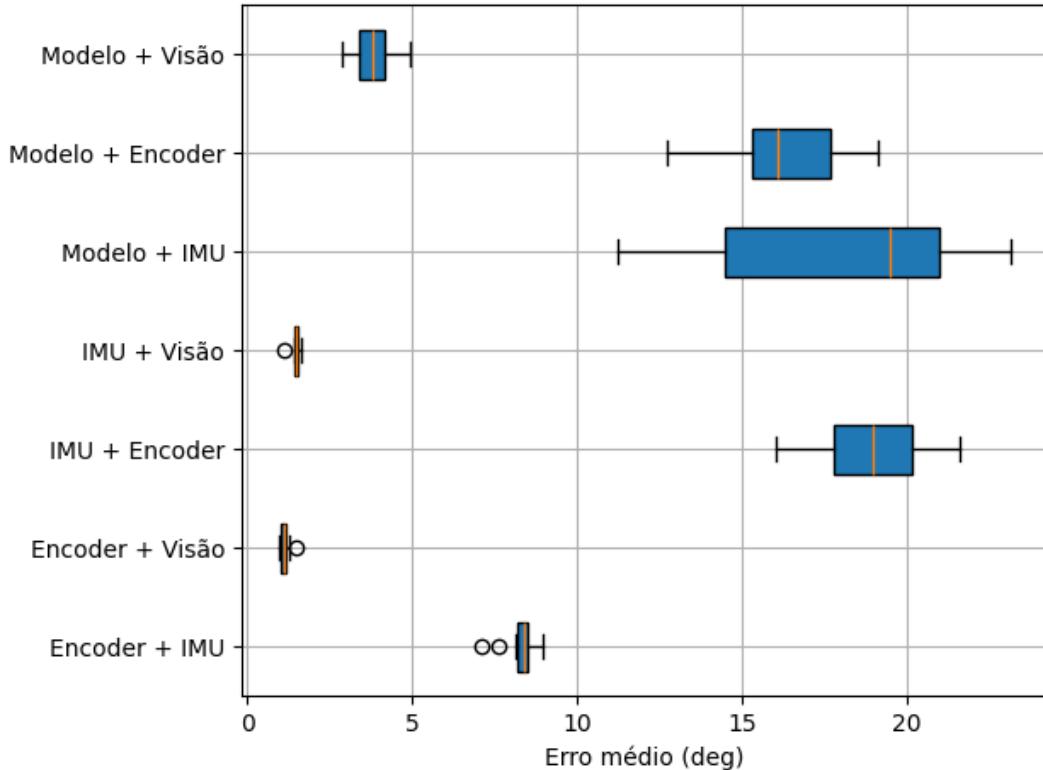
Fonte: o Autor

A Figura 38 mostra como a orientação calculada pelo EKF começa a desviar do que é lido pelo SSL-Vision no Cenário 5 (IMU + encoders), mostrando um acúmulo de erro ao longo do trajeto. Já na Figura 39, que mostra a orientação calculada pelo EKF no Cenário 7 (encoders + IMU), não é notável esse acúmulo ao longo do tempo, mesmo que os sensores utilizados sejam os mesmos.

Por fim, a Tabela 14 traz os resultados gerais obtidos na região de maior distorção por conta do efeito olho de peixe das câmeras com relação ao erro médio de distância, erro médio de orientação e o tempo de atualização médio. Quanto à posição, assim como no trajeto quadrado menor, os cenários que utilizam sensores na fase de predição (Cenários 4 e 6) junto do SSL-Vision na fase de correção apresentam resultados levemente melhores do que o sistema atual da equipe (Cenário 8) em relação à média, valor máximo e desvio padrão.

Quanto ao tempo de atualização da estimativa de posição, os resultados obtidos também seguem o que foi observado na Seção 6.3, em que o sistema atual utilizado pela equipe RoboFEI tem média praticamente 5 vezes superior aos cenários de fusão de sensores de forma embarcada.

Figura 37 – Erro médio de orientação no teste com trajeto do quadrado maior.



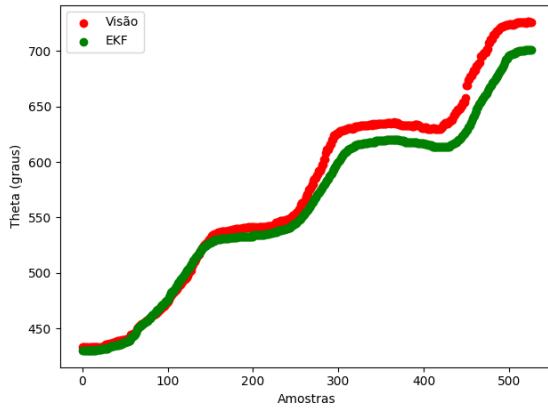
Fonte: o Autor

Esse resultado obtido com a fusão de sensores é essencial para um bom controle de posição, como descrito na Seção 2.1.2.

Tabela 14 – Resumo dos resultados obtidos dos cenários no teste com quadrado maior.

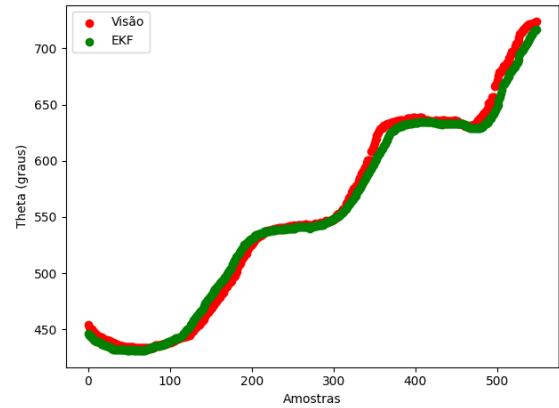
<b>Cenário</b>	Distância - Erro			Orientação - Erro		$T_{att}$ [ms]
	<b>Média</b> [m]	<b>Máximo</b> [m]	<b>Desv. Pad.</b> [m]	<b>Média</b> [deg]	<b>Máximo</b> [deg]	
Modelo + Visão	0,02586	0,062953	0,028476	3,824521	15,458832	3,47707
Modelo + Encoders	0,082803	0,149021	0,073858	16,23678	38,790607	3,58073
Modelo + IMU	0,277591	0,413911	0,222238	17,841146	45,34764	3,59174
IMU + Visão	0,015806	0,046374	0,019426	1,470293	5,637516	3,52665
IMU + Encoders	0,089906	0,155431	0,07952	18,94846	46,190327	3,59629
Encoders + Visão	0,015518	0,040223	0,01688	1,128545	3,986355	3,46823
Encoders + IMU	0,108431	0,174239	0,093185	8,268624	23,724647	3,59208
Visão	0,021699	0,048287	0,020637	-	-	16,4081

Figura 38 – Orientação ao longo de uma amostra do Cenário 5.



Fonte: o Autor

Figura 39 – Orientação ao longo de uma amostra do Cenário 7.



Fonte: o Autor

## 6.5 CONSIDERAÇÕES FINAIS

A partir do que foi analisado com os gráficos em cada um dos trajetos propostos para análise dos resultados, é possível notar que a utilização de um sensor que te dê informações absolutas do seu robô em relação ao ambiente é essencial para a fusão de sensores. Especialmente na SSL, que a precisão necessária para realização de jogadas é alta por conta da velocidade e tamanho tanto dos robôs, quanto da bola utilizada, utilizar o sistema de visão da liga é essencial para realizar a fusão de sensores para localização de forma embarcada.

Entretanto, é notável que a fusão de sensores é superior à utilização sozinha dos dados do sistema de visão, tanto por conta da leve melhoria e confiabilidade alcançada ao utilizar outros sensores junto ao SSL-Vision, quanto pelo tempo de atualização praticamente 5 vezes mais demorado do que o sistema de fusão de sensores. Como mostrado na Seção 2.1.2, um sistema de localização embarcado é essencial para realização de jogadas na SSL ao tornar viável a mudança do controle de posição dos robôs para seu microcontrolador ao invés do computador central utilizado pelas equipes.

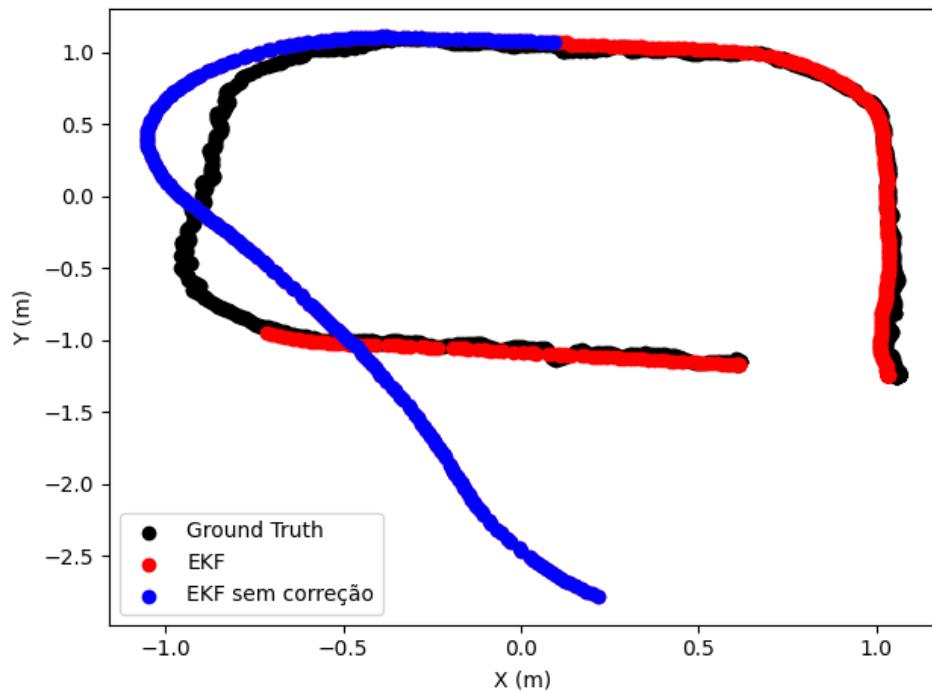
A partir do que foi analisado nos resultados obtidos nos testes em diferentes regiões do campo, descrito nas Seções 6.3 e 6.4, é notável que as melhores combinações foram nos Cenários 4 e 6, que utilizam os sensores IMU e encoders, respectivamente, junto aos dados do SSL-Vision para a fusão de sensores.

Um efeito que acontece durante partidas da SSL é a oclusão, seja da bolinha ou dos robôs, muito por conta de a iluminação não ser uniforme comparando diferentes regiões do campo.

Essas situações acontecem em sua maioria nas regiões perto do escanteio, ou próxima aos gols no campo, e durante curtos períodos de tempo normalmente (de 1 a 2 segundos, como piscadas), visto a dinâmica das partidas. Sabendo disso, decidiu-se testar as duas melhores combinações em situações até mais críticas do que essa descrita, no caso com o robô não sendo notado pelo sistema de visão durante 8 segundos no trajeto.

As Figuras 40 e 41 mostram os testes realizados desconsiderando os dados do SSL-Vision durante 8 segundos de trajeto para os Cenários 4 e 6, respectivamente. No caso, os pontos em vermelho são as posições calculadas pelo EKF que utilizam os dados do SSL-Vision para correção, enquanto os pontos em azul são os pontos que o sistema para de considerar os dados do SSL-Vision, performando apenas a fase de predição, já os pontos em preto são os pontos do sistema de *ground truth*.

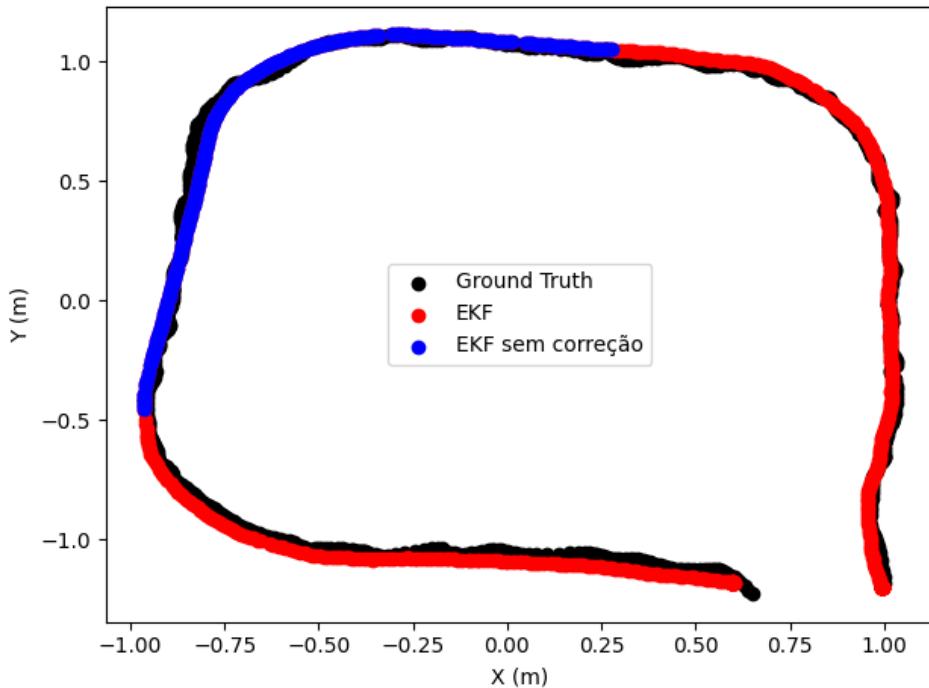
Figura 40 – Teste de 8 segundos sem visão para o Cenário 4.



Fonte: o Autor

É notável que a IMU consegue predizer bem a posição do robô até 2 segundos, mas após isso por conta da integração dos valores de aceleração para obter dados de posição e velocidade começa a aumentar significativamente o erro, e volta para o trajeto quando os dados da visão retornam. Enquanto isso, os encoders conseguem predizer bem a localização do robô durante esse tempo expressivo sem os dados do sistema de visão global, já que a maior causa de erros

Figura 41 – Teste de 8 segundos sem visão para o Cenário 6.



Fonte: o Autor

para as leituras do encoder é em momentos de aceleração ou desaceleração do robô por conta de derrapagens.

A partir dos testes sem o sistema de visão durante 8 segundos e dos resultados descritos nas Seções 6.3 e 6.4, é possível colocar que a utilização do encoder seja mais benéfica do que a utilização da IMU, visto que os resultados de ambas são parecidos quando há informação disponível do SSL-Vision, mas sem o sistema de visão há um erro acumulado grande em pouco tempo utilizando a IMU somente. Uma solução também pode ser, enquanto uma nova informação do SSL-Vision não estiver disponível, realizar a fusão dos encoders e IMU.

Entretanto, considerando que o sistema está inserido na lógica do SSL, o sistema de localização da equipe pode ser desenvolvido para lidar com as situações que o robô não é identificado pelo sistema de visão, como realizar a fusão entre encoder e IMU após um determinado tempo que o robô não seja identificado. Assim, os dados dos encoders evitariam o aumento de incerteza dos estados do EKF, diferente da utilização da IMU, que acumularia um erro em pouco tempo, mesmo que o robô esteja parado.

Caso não haja a possibilidade de utilização de um sensor que forneça informações globais, como LiDAR, câmera ou GPS, apenas a utilização de IMU e encoder pode trazer resultados

aceitáveis (abaixo de 10cm na média de um trajeto), ou até a utilização do modelo sendo corrigido pelos dados do encoder (Cenário 2). Porém, os resultados mostram que a utilização de um sensor global traz uma ótima precisão para um projeto de localização de um robô móvel.

## 7 CONCLUSÃO

Este trabalho realizou uma análise comparativa do uso de diferentes sensores num sistema de localização baseado em fusão de sensores de forma embarcada aplicado no ambiente da categoria SSL de futebol de robôs da RoboCup. Para realizar a fusão dos diferentes sensores foi escolhido utilizar um Filtro de Kalman Estendido, que é separado nas fases de predição e correção. Os sensores escolhidos foram os seguintes: encoders acoplados às rodas do robô, giroscópio e acelerômetro presentes na placa eletrônica do robô para medições de velocidade angular e aceleração linear, respectivamente, o sistema de câmeras da categoria SSL e, por fim, o modelo cinemático de um robô omnidirecional de 4 rodas.

Além da implementação do sistema de fusão de sensores, o sistema de *ground truth* também foi desenvolvido para esse projeto e é baseado nas leituras de dois LiDARs, cada um posicionado em um dos gols do campo. O sistema utiliza Transformada de Hough para identificação de um padrão circular na imagem dos pontos dos LiDARs. Os resultados obtidos mostraram a qualidade do sistema com erro médio próximo de 0,5cm em ambos os eixos do campo, apesar do sistema ser menos uniforme ao apresentar um desvio padrão maior que o sistema de câmeras da SSL.

Através do erro médio em relação ao sistema de *ground truth* e o tempo de atualização da estimativa de localização foi possível analisar de maneira aprofundada o desempenho dos sensores escolhidos em trajetos distintos para um sistema de localização de forma embarcada.

Os dois testes conduzidos visaram analisar as combinações propostas de fusão de sensores em situações características do sistema de câmeras, que no caso são a região de sobreposição de câmeras, normalmente situada no centro do campo e que leva a divergências na posição observada dos robôs por duplicações, e na região de distorção das câmeras, que normalmente acontecem nas bordas do campo por conta do efeito olho de peixe das lentes utilizadas e acaba influenciando na posição lida pelo sistema.

Os resultados obtidos demonstraram que as melhores combinações para os robôs da categoria SSL utilizam os dados de sensores na fase de predição do EKF e o sistema de visão da liga na fase de correção, apesar das deficiências demonstradas do sistema. Isso é notável analisando o erro médio em relação ao sistema de *ground truth*, assim como os valores máximos e os desvios padrões, dos Cenários 4 e 6 comparados com os outros. Os dados dos sensores provêm melhores estimativas de velocidade e aceleração do movimento do robô. Já a utilização do modelo cinemático do robô omnidirecional na fase de predição não se provou uma boa opção,

justificado por não levar em conta o atrito das rodas omnidirecionais, por exemplo, ou até mesmo não-linearidades envolvidas no movimento do robô.

Embora o SSL-Vision forneça a pose do robô a uma taxa de 60 frames por segundo aproximadamente, é importante que no intervalo entre frames o robô mantenha estimativas da sua posição devido a dinamicidade das partidas da SSL a fim de realizar chutes ou interceptações mais precisas. Portanto, um sistema de localização embarcado com um período de atualização em torno de 3.5ms é ideal para a realização de jogadas durante as partidas, como as combinações de fusão de sensores testadas nesse projeto.

Apesar de os testes estarem inseridos na lógica do futebol de robôs, especialmente na categoria SSL, os resultados obtidos nesse estudo podem beneficiar outras aplicações de robótica móvel, como robôs de serviço ou robôs humanoides. Por exemplo, apesar de os resultados obtidos nos cenários que não utilizam o sistema global de visão da liga não serem satisfatórios para a aplicação, acredita-se que os erros calculados sejam aceitáveis em aplicações de robôs maiores e mais robustos, e que até fazem utilização de outros sensores globais, como LiDARs em aplicações internas ou GPS em aplicações externas.

Trabalhos futuros relacionados estão ligados à implementação do sistema de localização de fusão de sensores testado nesse trabalho em um controle de posição dos robôs da categoria de modo embarcado, mostrando a efetividade do sistema de localização na malha de controle do robô a fim de melhorar a jogabilidade da equipe em campeonatos.

## REFERÊNCIAS

AGUIAR, João Victor Lourenço; DA SILVA COSTA, Leonardo; TONIDANDEL, Flavio. Linear Quadratic Regulator Path Tracking for Omnidirectional Robots in High-Dynamic Environments. In: 2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE). [S.l.: s.n.], 2023. P. 266–271. DOI: 10.1109/LARS/SBR/WRE59448.2023.10332958.

AGUIAR, João Victor Lourenço et al. Reformulação do Sistema de Deslocamento de Robôs Omnidirecionais. **Simpósio de Iniciação Científica, Didática e de Ações Sociais de Extensão da FEI**, p. 48, 2022.

AGUIAR, Luis et al. Kalman filtering for differential drive robots tracking. In: XIII Simposio Brasileiro de Automacao Inteligente,(October 1-4, 2017, Porto Alegre, 1520. [S.l.: s.n.], 2017.

ALATISE, Mary B; HANCKE, Gerhard P. A review on challenges of autonomous mobile robot and sensor fusion methods. **IEEE Access**, IEEE, v. 8, p. 39830–39846, 2020.

**ARM. CMSIS-DSP Library.** Acesso em 4 jan. 2025 as 19:06. Jul. 2024. Disponível em: [https://arm-software.github.io/CMSIS\\_6/main/DSP/index.html](https://arm-software.github.io/CMSIS_6/main/DSP/index.html).

BAIE, Chuan Yu et al. **2017 Team Description Paper: UBC Thunderbots.** [S.l.: s.n.], 2017. Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

**BCC. Sensors: Technologies and Global Markets.** Acesso em 06 jun. 2024 as 16:18. Mar. 2024. Disponível em: <https://www.bccresearch.com/market-research/instrumentation-and-sensors/sensors-technologies-markets-report.html>.

BEHZAD, Kian et al. **PARSIAN 2019: Extended Team Description Paper.** [S.l.: s.n.], 2019. Extended Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

BENARBIA, Taha; KYAMAKYA, Kyandoghere. A Literature Review of Drone-Based Package Delivery Logistics Systems and Their Implementation Feasibility. **Sustainability**, v. 14, n. 1, 2022. ISSN 2071-1050. DOI: 10.3390/su14010360. Disponível em: <https://www.mdpi.com/2071-1050/14/1/360>.

BORENSTEIN, Johann et al. Mobile robot positioning: Sensors and techniques. **Journal of robotic systems**, Wiley Online Library, v. 14, n. 4, p. 231–249, 1997.

BROOKS, Richard R; IYENGAR, Sundararaja S. **Multi-sensor fusion: fundamentals and applications with software.** [S.l.]: Prentice-Hall, Inc., 1998.

CASTANEDO, Federico et al. A review of data fusion techniques. **The scientific world journal**, Hindawi, v. 2013, 2013.

CHEN, Lingyun et al. **ZJUNlict Extended Team Description Paper for RoboCup 2018.** [S.l.: s.n.], 2018. Extended Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

CHURCHLEY, Scott et al. **2015 Team Description Paper: UBC Thunderbots.** [S.l.: s.n.], 2015. Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

COITO, Francisco et al. Tracking a Mobile Robot Position Using Vision and Inertial Sensor. In: CAMARINHA-MATOS, Luis M.; BARRENTO, Nuno S.; MENDONÇA, Ricardo (Ed.). **Technological Innovation for Collective Awareness Systems.** Berlin, Heidelberg: Springer Berlin Heidelberg, 2014. P. 201–208. ISBN 978-3-642-54734-8.

COSTA, Leonardo da Silva; TONIDANDEL, Flavio. **Análise de técnicas de navegação de robôs autônomos em ambientes dinâmicos e incertos.** Set. 2023. Dissertação de Mestrado – Centro Universitário FEI, São Bernardo do Campo.  
<https://repositorio.fei.edu.br/handle/FEI/5194>.

DADAFSHAR, Majid. Accelerometer and gyroscopes sensors: operation, sensing, and applications. **Maxim Integrated [online]**, 2014.

DASARATHY, Belur V. Sensor fusion potential exploitation-innovative architectures and illustrative applications. **Proceedings of the IEEE**, IEEE, v. 85, n. 1, p. 24–38, 1997.

DINGMAN, Jeremy. **What Is A Gyroscope?** Acesso em 16 mar. 2024 as 15:32. Set. 2020. Disponível em: [https://aerospace.honeywell.com/us/en/about-us/blogs/what-is-a-gyroscope?utm\\_source=google%5C&utm\\_medium=cpc%5C&utm\\_campaign=23-aero-ww-dsa-blogs%5C&utm\\_content=dyn-en-lp%5C&gad\\_source=1%5C&gclid=Cj0KCQjwwMqvBhCtARIxAIXsZpb80B5mORQKLaGBbWRu0O5pSudNu1Xfje7I3ofgDTiBVRPTMROtyR4aAiK-EALw\\_wcB](https://aerospace.honeywell.com/us/en/about-us/blogs/what-is-a-gyroscope?utm_source=google%5C&utm_medium=cpc%5C&utm_campaign=23-aero-ww-dsa-blogs%5C&utm_content=dyn-en-lp%5C&gad_source=1%5C&gclid=Cj0KCQjwwMqvBhCtARIxAIXsZpb80B5mORQKLaGBbWRu0O5pSudNu1Xfje7I3ofgDTiBVRPTMROtyR4aAiK-EALw_wcB).

DOORNKAMP, Cas et al. RoboTeam Twente 2018 Team Description Paper. In.

DURRANT-WHYTE, Hugh F. Sensor models and multisensor integration. **The international journal of robotics research**, Sage Publications Sage CA: Thousand Oaks, CA, v. 7, n. 6, p. 97–113, 1988.

DYER, Benjamin M. et al. Filtering Strategies for State Estimation of Omniwheel Robots. In: 2020 IEEE International Conference on Mechatronics and Automation (ICMA). [S.l.: s.n.], 2020. P. 186–191. DOI: 10.1109/ICMA49215.2020.9233826.

ELMENREICH, Wilfried. An introduction to sensor fusion. **Vienna University of Technology, Austria**, v. 502, p. 1–28, 2002.

EMAN, Alhamdi; RAMDANE, Hedjar. Mobile robot localization using extended Kalman filter. In: IEEE. 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS). [S.l.: s.n.], 2020. P. 1–5.

FUNG, Man Lok; CHEN, Michael ZQ; CHEN, Yong Hua. Sensor fusion: A review of methods and applications. In: IEEE. 2017 29th Chinese Control And Decision Conference (CCDC). [S.l.: s.n.], 2017. P. 3853–3860.

GONÇALVES, José; LIMA, José; COSTA, Paulo Gomes da. Real-time tracking of an omnidirectional robot: an extended kalman filter approach. In: INTERNATIONAL Conference on Informatics in Control, Automation and Robotics. [S.l.: s.n.], 2008.

HALL, David L; LLINAS, James. An introduction to multisensor data fusion. **Proceedings of the IEEE**, IEEE, v. 85, n. 1, p. 6–23, 1997.

HASSAN, Mahmood ul; BAO, Qilian. A field calibration method for low-cost mems accelerometer based on the generalized nonlinear least square method. **Multiscale Science and Engineering**, Springer, v. 2, p. 135–142, 2020.

HASSANEIN, Allam Shehata et al. A survey on Hough transform, theory, techniques and applications. **arXiv preprint arXiv:1502.02160**, 2015.

HUANG, Zheyuan et al. ZJUNlct Extended Team Description Paper for Robocup 2020. In.

ISMAIL, Muhammad Azhar; PURWANTO, Djoko; ARIFIN, Achmad. Soccer Robot Localization Based on Sensor Fusion From Odometry and Omnidvision. In: IEEE. 2022 International Seminar on Intelligent Technology and Its Applications (ISITIA). [S.l.: s.n.], 2022. P. 273–278.

JAIN, A.; ZHANG, L.; JIANG, L. **High-Fidelity Sensor Calibration for Autonomous Vehicles**. Acesso em 06 jun. 2024 as 16:31. Ago. 2019. Disponível em: <https://medium.com/wovenplanetlevel5/high-fidelity-sensor-calibration-for-autonomous-vehicles-6af06eba4c26>.

KHODARAHMI, Masoud; MAIHAMI, Vafa. A review on Kalman filter models. **Archives of Computational Methods in Engineering**, Springer, v. 30, n. 1, p. 727–747, 2023.

KOROTAJ, Blaž; NOVOSELNIK, Branimir; BAOTIĆ, Mato. Kalman filter based sensor fusion for omnidirectional mechatronic system. In: IEEE. 2021 International Conference on Electrical Drives & Power Electronics (EDPE). [S.l.: s.n.], 2021. P. 183–188.

LI, Xianzhi et al. Data fusion for intelligent crowd monitoring and management systems: A survey. **IEEE Access**, IEEE, v. 9, p. 47069–47083, 2021.

LOGITECH. **BRIO ULTRA HD BUSINESS WEBCAM**. [S.l.], mar. 2021.

LOTUFO, Roberto A et al. Morphological image processing. In: MICROSCOPE image processing. [S.l.]: Elsevier, 2023. P. 75–117.

LUIZ R., José. **Como funciona un Encoder**. Acesso em 26 mar. 2024 as 17:13. Mar. 2021. Disponível em: <https://como-funciona.co/un-encoder/>.

LUO, Ren C; CHANG, Chih Chia; LAI, Chun Chi. Multisensor fusion and integration: Theories, applications, and its perspectives. **IEEE Sensors Journal**, IEEE, v. 11, n. 12, p. 3122–3138, 2011.

LUO, Ren C; KAY, Michael G. A tutorial on multisensor integration and fusion. In: IEEE. [PROCEEDINGS] IECON'90: 16th Annual Conference of IEEE Industrial Electronics Society. [S.l.: s.n.], 1990. P. 707–722.

LV, Jiajun et al. Targetless calibration of lidar-imu system based on continuous-time batch estimation. In: IEEE. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). [S.l.: s.n.], 2020. P. 9968–9975.

MACKWORTH, Alan K. On Seeing Robots, 1982.

MÁRTON, Lőrinc; GYÖRGY, Katalin. Two-Stage Kalman Filtering for Indoor Localization of Omnidirectional Robots. **Electrical and Mechanical Engineering**, v. 5, p. 44–60, 2013.

MAXON GROUP. **EC 45 brushless 50W with sensor hall**. [S.l.], abr. 2019.

MELO, João G; BARROS, Edna. An embedded monocular vision approach for ground-aware objects detection and position estimation. In: ROBOT World Cup. [S.l.]: Springer, 2022. P. 100–111.

MENDES, Jorge Miguel Ferreira da Silva. Perceção visual semântica de uma vinha para auxílio à navegação de robôs, 2017.

MENDES JR., José Jair Alves et al. Sensor Fusion and Smart Sensor in Sports and Biomedical Applications. **Sensors**, v. 16, n. 10, 2016. ISSN 1424-8220. DOI: 10.3390/s16101569. Disponível em: <https://www.mdpi.com/1424-8220/16/10/1569>.

MENEZES FILHO, Rogério P et al. Triaxial accelerometer calibration using an extended two-step methodology. In: IEEE. 2020 Latin American Robotics Symposium (LARS), 2020 Brazilian Symposium on Robotics (SBR) and 2020 Workshop on Robotics in Education (WRE). [S.l.: s.n.], 2020. P. 1–6.

MUKHOPADHYAY, Priyanka; CHAUDHURI, Bidyut B. A survey of Hough Transform. **Pattern Recognition**, Elsevier, v. 48, n. 3, p. 993–1010, 2015.

NAEINI, Meisam Kasaeian et al. **MRL Extended Team Description 2020**. [S.l.: s.n.], 2020. Extended Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

NAGLA, KS; UDDIN, Moin; SINGH, Dilbag. Multisensor data fusion and integration for mobile robots: A review. **IAES International Journal of Robotics and Automation**, IAES Institute of Advanced Engineering e Science, v. 3, n. 2, p. 131, 2014.

NEGENBORN, R.R. **Robot Localization and Kalman Filters. On finding your position in a noisy world**. Jan. 2003. Tese (Doutorado).

NIETO, Juan; BAILEY, Tim; NEBOT, Eduardo. Recursive scan-matching SLAM. **Robotics and Autonomous Systems**, v. 55, n. 1, p. 39–49, 2007. Simultaneous Localisation and Map Building. ISSN 0921-8890. DOI: <https://doi.org/10.1016/j.robot.2006.06.008>. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0921889006001461>.

NISTLER, Jonathan R.; SELEKWA, Majura F. Gravity compensation in accelerometer measurements for robot navigation on inclined surfaces. **Procedia Computer Science**, v. 6, p. 413–418, 2011. Complex adaptive sysytems. ISSN 1877-0509. DOI: <https://doi.org/10.1016/j.procs.2011.08.077>. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1877050911005424>.

NORDIC SEMICONDUCTOR. **nRF24L01 Single Chip 2.4GHz Transceiver - Product Specification**. [S.I.], jul. 2007. Rev. 2.0.

OLIVEIRA, Bruno Queres de et al. Tipos e aplicações de sensores na robótica. **Caderno de Graduação-Ciências Exatas e Tecnológicas-UNIT-ALAGOAS**, v. 4, n. 1, p. 223–223, 2017.

PACHECO, Marcus Vinicius O.; SILVA, Felipe O.; FARRELL, Jay A. GPS-Aided Odometry Navigation for IARs: Comparison Between Loosely and Tightly Coupled Integrations Under Restricted Satellite Visibility Conditions. In: 2023 Latin American Robotics Symposium (LARS), 2023 Brazilian Symposium on Robotics (SBR), and 2023 Workshop on Robotics in Education (WRE). [S.I.: s.n.], 2023. P. 278–283. DOI: [10.1109/LARS/SBR/WRE59448.2023.10333060](https://doi.org/10.1109/LARS/SBR/WRE59448.2023.10333060).

PANIGRAHI, Prabin Kumar; BISOY, Sukant Kishoro. Localization strategies for autonomous mobile robots: A review. **Journal of King Saud University - Computer and Information Sciences**, v. 34, 8, Part B, p. 6019–6039, 2022. ISSN 1319-1578. DOI: <https://doi.org/10.1016/j.jksuci.2021.02.015>. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1319157821000550>.

PAPAFOTIS, Konstantinos; SOTIRIADIS, Paul P. Exploring the Importance of Sensors' Calibration in Inertial Navigation Systems. In: 2020 IEEE International Symposium on Circuits and Systems (ISCAS). [S.I.: s.n.], 2020. P. 1–4. DOI: [10.1109/ISCAS45731.2020.9181212](https://doi.org/10.1109/ISCAS45731.2020.9181212).

PAREKH, Darsh et al. A review on autonomous vehicles: Progress, methods and challenges. **Electronics**, MDPI, v. 11, n. 14, p. 2162, 2022.

PASSARO, Vittorio M. N. et al. Gyroscope Technology and Applications: A Review in the Industrial Perspective. **Sensors**, v. 17, n. 10, 2017. ISSN 1424-8220. DOI: [10.3390/s17102284](https://doi.org/10.3390/s17102284). Disponível em: <https://www.mdpi.com/1424-8220/17/10/2284>.

PAULI, Guilherme; TONIDANDEL, Flavio. Solving the Time Lapse from Vision System in a Robot Soccer Game Using Kalman Filter. In: 2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE). [S.I.: s.n.], 2019. P. 7–12. DOI: [10.1109/LARS-SBR-WRE48964.2019.00010](https://doi.org/10.1109/LARS-SBR-WRE48964.2019.00010).

PERSSON, Anders. How do we understand the Coriolis force? **Bulletin of the American Meteorological Society**, American Meteorological Society, v. 79, n. 7, p. 1373–1386, 1998.

RAJ, Ravi; KOS, Andrzej. A Comprehensive Study of Mobile Robot: History, Developments, Applications, and Future Research Perspectives. **Applied Sciences**, v. 12, n. 14, 2022. ISSN 2076-3417. DOI: 10.3390/app12146951. Disponível em: <https://www.mdpi.com/2076-3417/12/14/6951>.

RIGATOS, Gerasimos G. Extended Kalman and particle filtering for sensor fusion in motion control of mobile robots. **Mathematics and computers in simulation**, Elsevier, v. 81, n. 3, p. 590–607, 2010.

ROBOCUP. **A Brief History of RoboCup**. Acesso em 13 mar. 2024 as 14:00. 2020. Disponível em: [https://www.robocup.org/a\\_brief\\_history\\_of\\_robocup](https://www.robocup.org/a_brief_history_of_robocup).

ROBOCUP-SSL. **About the Small Size League**. Acesso em 28 abr. 2024 as 18:18. Set. 2019. Disponível em: <https://ssl.robocup.org/about/>.

ROBOCUP-SSL. **Página das regras da SSL**. Acesso em 1 mar. 2024 as 21:33. Set. 2019. Disponível em: <https://ssl.robocup.org/rules/>.

ROSTAMI, Vahid et al. Localization and Navigation Omni-directional Robots based on Sensors Fusion and Particle Filter. In: IEEE. 2018 9th Conference on Artificial Intelligence and Robotics and 2nd Asia-Pacific International Symposium. [S.l.: s.n.], 2018. P. 65–73.

RYLL, Andre et al. **TIGERS Mannheim (Team Interacting and Game Evolving Robots) Team Description for RoboCup 2013**. [S.l.: s.n.], 2013. Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

RYLL, Andre et al. TIGERS Mannheim Team Description for RoboCup 2013. In. Disponível em: <https://api.semanticscholar.org/CorpusID:202684288>.

SAE INTERNATIONAL. On-Board System Requirements for V2V Safety Communications. **SAE J2945/1**, 2016.

SALEHI, Ali et al. **Immortals 2023 Extended Team Description Paper**. [S.l.: s.n.], 2023. Extended Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

SANTINI, A; NICOSIA, S; NANNI, V. Trajectory estimation and correction for a wheeled mobile robot using heterogeneous sensors and Kalman filter. **IFAC Proceedings Volumes**, Elsevier, v. 30, n. 20, p. 11–16, 1997.

SASIADEK, J.Z.; HARTANA, P. Sensor data fusion using Kalman filter. In: PROCEEDINGS of the Third International Conference on Information Fusion. [S.l.: s.n.], 2000. v. 2, wed5/19–wed5/25 vol.2. DOI: 10.1109/IFIC.2000.859866.

SIEGWART, R.; NOURBAKHSH, I.R.; SCARAMUZZA, D. **Introduction to Autonomous Mobile Robots, second edition**. [S.l.]: MIT Press, 2011. (Intelligent Robotics and Autonomous Agents series). ISBN 9780262015356. Disponível em: <https://books.google.com.br/books?id=4of6AQAAQBAJ>.

SINGHAL, Amit. Issues in autonomous mobile robot navigation. **Computer Science Dept, U. of Rochester**, Citeseer, p. 74, 1997.

SLAMTEC. **RPLIDAR A1 - Low Cost Degree Laser Range Scanner**. [S.l.], jul. 2016. Rev. 10.

SLAMTEC. **RPLIDAR S2 - Low Cost 360 Degree Laser Range Scanner**. [S.l.], mar. 2021. Rev. 1.7.

STATISTA. **Robotics - Worldwide**. Acesso em 09 jun. 2024 as 22:54. Ago. 2023. Disponível em: <https://www.statista.com/outlook/tmo/robotics/worldwide?currency=USD#revenue>.

STMICROELECTRONICS. **Discovery kit with STM32F411VE MCU - User manual**. [S.l.], set. 2020. Rev. 2.

STMICROELECTRONICS. **MEMS motion sensor: three-axis digital output gyroscope**. [S.l.], abr. 2015. Rev. 2.

STMICROELECTRONICS. **Repositório do driver do componente I3G4250D**. Acesso em 20 mar. 2024 as 20:07. Dez. 2023. Disponível em: <https://github.com/STMicroelectronics/stm32-i3g4250d>.

STMICROELECTRONICS. **Repositório do driver do componente LSM303AGR**. Acesso em 20 mar. 2024 as 18:58. Dez. 2023. Disponível em: <https://github.com/STMicroelectronics/stm32-lsm303agr>.

STMICROELECTRONICS. **Ultra-compact high-performance eCompass module: 3D accelerometer and 3D magnetometer**. [S.l.], ago. 2022. Rev. 11.

SULIMAN, Caius; CRUCERU, Cristina; MOLDOVEANU, Florin. Mobile robot position estimation using the Kalman filter. **Acta Marisiensis. Seria Technologica**, De Gruyter Poland, v. 6, p. 75, 2009.

TECHNEXION. **Vision-guided Robotics - How Cameras are Transforming Robotics**. Acesso em 19 mar. 2024 as 20:14. Abr. 2023. Disponível em: <https://www.technexion.com/resources/vision-guided-robotics-how-cameras-are-transforming-robotics/#:~:text=Role%20of%20a%20Camera%20in%20Perception%20Enhancement&text=Robots%20can%20learn%20much%20about,their%20environment%20in%20greater%20depth..>

TECHNOLOGIES, Allied Vision. **Technical Manual**. [S.l.], abr. 2011. V4.4.2.

THRUN, Sebastian. Probabilistic robotics. **Communications of the ACM**, ACM New York, NY, USA, v. 45, n. 3, p. 52–57, 2002.

THRUN, Sebastian et al. Robust Monte Carlo localization for mobile robots. **Artificial intelligence**, Elsevier, v. 128, n. 1-2, p. 99–141, 2001.

URREA, Claudio; AGRAMONTE, Rayko. Kalman filter: historical overview and review of its use in robotics 60 years after its creation. **Journal of Sensors**, Wiley Online Library, v. 2021, n. 1, p. 9674015, 2021.

US DIGITAL. E4T Miniature Optical Kit Encoder. [S.l.], jan. 2024.

VAKIL, Asad et al. A survey of multimodal sensor fusion for passive RF and EO information integration. **IEEE Aerospace and Electronic Systems Magazine**, IEEE, v. 36, n. 7, p. 44–61, 2021.

VISSE, A; GROEN, FCA. Organisation and design of autonomous systems. Textbook, Faculty of Mathematics. **Computer Science, Physics and Astronomy, University of Amsterdam, Kruislaan**, v. 403, 1999.

WANG, Li et al. An Efficient Calibration Method for Triaxial Gyroscope. **IEEE Sensors Journal**, v. 21, n. 18, p. 19896–19903, 2021. DOI: 10.1109/JSEN.2021.3100589.

WELCH, Greg; BISHOP, Gary et al. An introduction to the Kalman filter. Chapel Hill, NC, USA, 1995.

WHITE, Franklin E et al. Data fusion lexicon. **Joint Directors of Laboratories, Technical Panel for C**, v. 3, p. 19, 1991.

ZIBETTI, Andre. **Distribuição Normal (Gaussiana)**. Acesso em 23 jun. 2024 as 15:45. Jul. 2022. Disponível em: <https://www.inf.ufsc.br/~andre.zibetti/probabilidade/normal.html>.

ZICKLER, Stefan et al. SSL-Vision: The Shared Vision System for the RoboCup Small Size League. In: BALTES, Jacky et al. (Ed.). **RoboCup 2009: Robot Soccer World Cup XIII**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. P. 425–436. ISBN 978-3-642-11876-0.

ZOLANVARI, Alireza et al. **PARSIAN Team Description for RoboCup 2015**. [S.l.: s.n.], 2015. Team Description Paper, RoboCup Small Size League. Disponível em: <https://ssl.robocup.org/team-description-papers/>.

**ANEXO A – ARTIGO PUBLICADO NO SIMPÓSIO CROS 2025**

# Position Ground Truth For a Round Robot Using LiDARs

1<sup>st</sup> João Victor Lourenço Aguiar  
*Department of Electrical Engineering*  
*Centro Universitário FEI*  
São Bernardo do Campo, Brazil  
0009-0009-2544-0161

2<sup>nd</sup> Flavio Tonidandel  
*Department of Computer Science*  
*Centro Universitário FEI*  
São Bernardo do Campo, Brazil  
0000-0003-0345-668X

**Abstract**—Localization is an essential role a robot must have, as it must be autonomous and require less human intervention when operating in society. When developing and testing a localization system, it's crucial to compare what it calculates with a ground truth provided by a sensor not being used in the system developed. Thus, this paper proposes a ground truth approach using two LiDARs to detect with reasonable accuracy the actual position of round robots in the real world, like the ones used in the Small Size League of RoboCup. The league is a considerable scenario for tests since it uses a particular camera system positioned above the field to detect the robot's position, and the internal sensors, like encoders, cannot provide high accuracy due to wheel slippage. The software architecture includes a scan matching step to combine LiDARs, and the detection step is made through the Hough transform to recognize circular patterns in the image created with the LiDAR points. The results showed that the system based on the LiDARs is significant as ground truth, with a mean error of less than 1 centimeter in the 17 positions analyzed in tests and a standard deviation of around 1 centimeter. The results are notable when compared with the league's camera system in these scenarios. The system can be adapted to different sizes of robots and be used in any indoor environment, although the Small Size League was the main scenario for the tests.

**Index Terms**—Ground Truth, Position, LiDAR, Round Robots.

## I. INTRODUCTION

The fast development of technology has ushered in a new era of automation, in which robots are playing a pivotal role in changing industries and the everyday lives of people in several sectors, such as healthcare, manufacturing, agriculture, and even personal assistance. The increase in the use of robots requires more autonomy from them, with the ability to operate less with human intervention and control, such as moving and reaching a goal in an environment, but also more safety when interacting with people.

Localization is an important role that a robot must have to work autonomously in an environment and is a substantial research topic in the robotics field [3], [5], [16]. It's crucial to compare the estimation calculated by these to a ground truth, i.e., the factual localization of the robot in an environment, to

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001

979-8-3315-5288-6/25/\$31.00 ©2025 IEEE

evaluate different localization systems [14]. Besides that, using a reliable ground truth is essential to calibrate a localization system being developed, such as a sensor fusion used for robots localization for indoor environments, where using a GPS for ground truth cannot be considered, for example.

RoboCup is an organization with a long-term goal to have a team of humanoid autonomous robots by the middle of the 21st century that will play against the winner of the Human World Cup. This goal has been proposed with the idea of improving and promoting research in robotics [18]. The Small Size League (SSL) is one of the oldest leagues of RoboCup and aims to solve the problem of intelligent multi-agent cooperation in a high-dynamic environment of a soccer game with an orange golf ball on a green carpeted field [19].

The centralized position system provided by a camera pair mounted 6m above the field is a league particularity about the others. The shared vision system for the SSL, which is called SSL-Vision [26], processes images captured by the cameras. One of the objectives of the system proposal was to prevent each team from mounting its structure on the field. The system uses color segmentation, and the colored pattern detection above the robots showed good precision. However, the SSL-Vision cannot be considered a ground truth when used in a localization system, such as a Kalman filter for position estimation with the cameras providing measurements to correct the robots' location. Besides that, other sensors used in the SSL robot, like encoders attached to the wheels, cannot provide the position ground truth due to wheel slippage when accelerating, for example.

In real systems, such as mobile robots, determining the ground truth is nearly impossible. Except for certain specific cases and applications, the ground truth for positioning should be seen as a reliable approximation that is more accurate than other available options, rather than an exact representation of the robot's true position.

In this paper, we will present and evaluate the use of LiDARs to create a trustable and efficient ground truth system, especially for round robots, such as the ones used in the Small Size League of the RoboCup, shown in Fig. 1. Because of the high-dynamic environment that they are in, and the high precision needed in positioning to complete plays and score goals, there is a high need for ground truth since the camera

system used in the league may have noise and struggle with light interference or overlap regions when using two or more cameras [1], [2], [7]. So, the camera system of the league can't be considered as ground truth, as the data provided from it may be being used in a positioning system, and internal sensors don't provide enough accuracy for it.



Fig. 1: Small Size League robot example.

Section II presents the related works about the ground truth theme in the RoboCup scene and other applications as well. Section III explains both the hardware architecture, i.e., the disposal of the lasers in the field test, and the software architecture, which contains the method applied to detect the robot's position and how the laser data is employed. Section IV describes the methodology used to analyze the system quality. Section V will present the results achieved by the tests. Finally, Section VI presents the conclusions of the comparison for position ground truth between the LiDARs system and the camera system used in the SSL.

## II. RELATED WORKS

The ground truth problem has been addressed through the years using different sensors and approaches [8], [11], [12], [17], [23]. Despite that, most parts of the ground truth systems use different types of cameras, and the smallest part of them use LiDARs but do not combine the data of multiple lasers.

[17] presents an approach applied to the Standard Platform League (SPL) from RoboCup. In this paper, the authors used four Kinects along the sidelines of the field to achieve a reliable ground truth for the humanoid robots of the league. The foreground mask extraction uses RGB and depth information in each frame. Then, using the created mask, the player's positions are obtained after a data fusion re-projects all the detected positions by the Kinects. The results were obtained by monitoring the position of the players using LiDAR, achieving a minimum error of 8mm and a maximum error of 476mm.

Following in the approaches used in the SPL, [8] also developed a system using a Kinect. The sensor driver from ROS provides the XYZ-RGB point cloud, which is applied Euclidean clustering to obtain one cluster per robot in the field, and then the position acquired is transformed to the global coordinates of the field. The authors obtained the results following the methodology described in this paper, i.e., placing the robot in known positions and capturing the system's output

during 20 measurements. The authors achieved an average error of 10.41cm between the analyzed robot orientations.

[11] brings another approach developed especially for the SPL, but using one LiDAR to achieve the ground truth needed. In this case, the detection can be used for any object type, which is different from this paper, which leans into detecting particularly round robots. Besides that, the authors used only a laser for detection, but they commented that detecting points far from the laser position is challenging and the accuracy is worse. The authors suggest a multi-sensor approach to enhance the accuracy, which this paper tested since the detection uses two lasers on each field's half.

About examples out of the RoboCup field, [23] brings a low-cost distributed system of visual localization of mobile robots using cameras. The authors used six cameras above a polygon where the robot would traffic, and each camera zone is marked with Apriltag markers, and the robot is also marked with an Apriltag for its detection. The authors achieved an accuracy lower than 2cm in all the scenarios studied by changing the image resolution. Besides, the authors also analyzed the probability of detecting the robot marker, which depends on the marker size and the image resolution. The work differs from this paper in camera use since the capture of the polygon range uses six of it, besides adding an external marker in the robot for position detection by the system.

Finally, [12] presented a ground truth system developed especially for underwater benchmarking using a stereo camera pair and an external host computer for image processing and data recording. The paper compared two methods to detect the robot's position: template matching and color-based segmentation. The results showed a mean error higher than 2cm in all the axes (X, Y, and Z), besides the standard deviation, which is higher than 0.9cm. The paper differs from this on camera use, which leads to a high dependence on the environment illumination for better detection, which is an irrelevant problem when using LiDARs for detection.

## III. ROBOT DETECTION

### A. Hardware architecture

The robot detection has been made using an RPLIDAR A1 [20] and an RPLIDAR S1 [22] in a test field of 4.5m long by 3.5m wide, each of them positioned in the center of a goal, as it can be seen in Fig. 2, where the A1 is placed on the left goal, while the S1 is on the right side.

The choice of this setting derives from initial tests made, besides the physical limitations of the lasers, since the sensors' maximum distance range is 6m [20]. So, if the sensors were placed on the field's corners, for example, the sensors could lose essential data depending on the robot's position. Each sensor was mounted on a base with a 5cm height to ensure that they could capture centralized data of the robot and avoid problems with distortions capturing the robot's wheels. However, since the objective is to evaluate positioning systems and there is no plan to be used directly in a game, it is admissible that the LiDARs can be placed further into the field

depending on the field size, the path planned for the tests, or the LiDAR used.

Compared to the traditional cameras setup used to detect the positions of the robots in the field in the SSL, this setup is easier to build, since it does not require the sensors to be placed in the ceiling and/or considerable height.

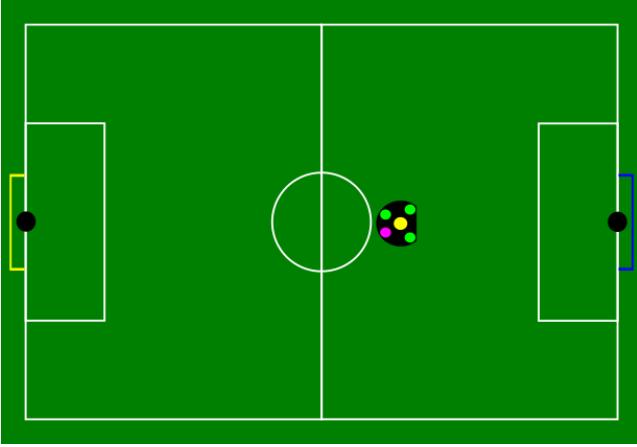


Fig. 2: Field test representation.

### B. Software architecture

Fig. 3 shows the software architecture of the system proposed. That is based on a ROS2 package that will apply a scan matching to find the relative position between both lasers. After the system merges the data of both LiDARs in an image, different morphological operation process the image in order to improve the image quality, proceed with the image detection to recognize the circular shape on it, and then calculate the global coordinate of the robot, since the system calculates the position in the LiDAR frame in a first moment.

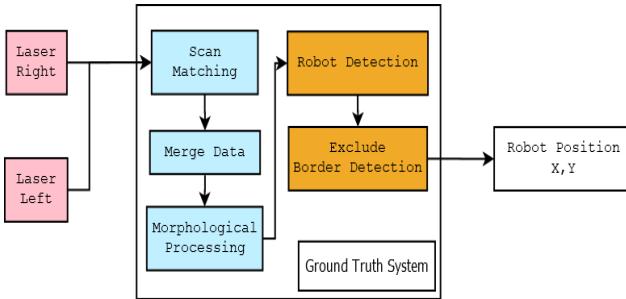


Fig. 3: Software architecture.

The first stage of the software is using scan matching made to find a transformation (translation, rotation, and scale) between both point clouds created by the lasers that fit better the data of both [15]. This stage is decisive since any centimeter difference between both data could lead to difficulty finding the robot's circular pattern or could add an offset in the position detected.

Basically, the scan matching works with two point clouds, the source P and the target S. The goal of the algorithm is to

find a transformation  $T = (R, t)$ , composed by a rotation R and translation t, that minimizes the difference between both point clouds P and S, as described in (1), where  $r_i$  is the nearest point in the target point cloud S.

$$T^* = \arg \min \sum_{i=1}^n \| (Rp_i + t) - r_i \|^2 \quad (1)$$

So, the following stage of the software is merging the data of both LiDARs in an image mask, that will be called laser points image, by applying the transformation found in the scan matching stage. To obtain the sensors' data, SLAMTEC ROS2 Package [21] for the LiDAR A1 and S1 was used, given that it is an easier way to communicate with the sensors, initialize them, and receive the data. The ROS2 node publishes a laser scan message that contains the ranges captured by the sensor and their angles. With these data, it's possible to build the laser points image following (2), which shows how to determine the coordinates of each object detected in a scan, where  $r$  is the range in meters and  $\theta$  is the respective angle to the range in radians.

$$\begin{cases} x = r * \cos(\theta) \\ y = r * \sin(\theta) \end{cases} \quad (2)$$

After merging the sensors' data, there is a layer to process the laser points image before recognizing the circular shape in the next stage. Morphological image processing is essential to refine the quality of images, reduce noise, and enhance the structural analysis, which leads to more accurate feature extraction on images [10]. First, the algorithm applies a subtle dilation to connect nearby points and then erodes them for better shape detection in the image. Finally, a delicate blur is applied to the laser points image for smoothness, thus reducing the noise in the image and the detection of false circles, as well as helping the pattern detection.

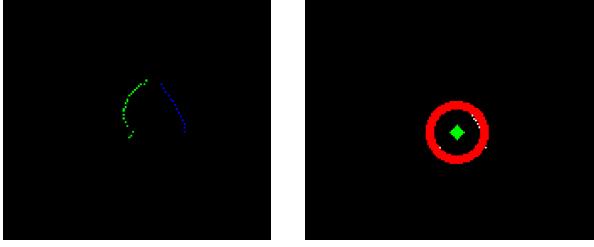
The next step of the software architecture is robot detection, that performs a circular or semi-circular detection in the laser points image. The method used for circular pattern detection is the Hough transform (HT). The technique was first created to detect lines in binary images, but other variations were suggested to detect different shapes, such as circular ones [13].

The circular HT relies on the circle equation demonstrated in (3), which the system calculates for each edge point in a pre-processed image according to the maximum and minimum limits for the radius. Each edge point contributes with votes for possible circles that can be part of. Thus, the votes are accumulated, and the local maxima provide the center locations and the radius of the circles in the images [6].

$$(x - a)^2 + (y - b)^2 = r^2 \quad (3)$$

In Fig. 4, it is possible to see two moments of the architecture. Fig. 4a shows the LiDARs' points after the scan matching and merge data stages, but before the morphological processing step, the blue points are from the right laser, and the green points are from the left laser. Fig. 4b shows the robot

detection step, where the Hough transform is applied to the laser points image to detect a circular pattern.



(a) Raw image of the LiDARs points.  
(b) Circle detected by the system.

Fig. 4: Two steps of the software architecture.

A last step of post processing is done to detect and exclude false positives detected in the field borders since most parts of the points detected by the laser are concentrated on field boundaries. Basically, the frontier coordinates are excluded and the detection is focused within the field.

#### IV. METHODOLOGY

This section explains the tests used to evaluate the system, the robot used during the tests, and the metrics analyzed to proof the ground truth system proposed in this paper.

As described previously, the robot used in the tests is one used in official matches of the SSL, so it fits in a 180x150mm cylinder, but it is linearly cut off on the front side to allocate the robot's dribbler system to control the ball in the game. Besides, the field used for the tests is 4.5x3.5m.

The vision system used in the tests is composed by two different cameras, each one positioned in each field's half. The cameras used are listed below. The first is placed on the right half of the field and the second is positioned on the left.

- Logitech BRIO 4K UHD [9]
- Stingray F046C [24] + Tamron 12VM412ASIR Lens

For the LiDARs system an RPLIDAR A1 [20] and an RPLIDAR S1 [22] were used in this paper, both of which are low-cost 360-degree LiDARs. The first uses the laser triangulation ranging principle [4] to achieve the data, while the second is based on the time-of-flight technology [25]. Both were operating with a scan frequency of 10Hz, and the A1 was used in Boost scan mode (787 points captured), while the S1 was in DenseBoost scan mode (925 points captured).

The notebook used to perform the robot detection in the laser points image has the following specifications:

- Intel i7-7700HQ CPU @ 2.80GHz
- NVIDIA Geforce MX150
- 16GB RAM
- Ubuntu 22.04.5 LTS x86\_64

In order to evaluate the proposed ground truth system using LiDARs, the robot will be placed in 17 known positions in the field, while the position is obtained by the LiDARs system and the vision system of the league. Both systems will be

compared with the real coordinate, taken using a measuring tape to obtain a better precision.

For each position the LiDARs system output will be recorded for 200 detections, while the data provided by the vision system is also being recorded. Then, the mean of these samples will be taken and compared with the known position to calculate the error, as well as their standard deviation will also be calculated in order to analyze the system stability.

The 17 scenarios are divided into two sets. First, the scenarios placed more centralized with the field coordinates (4-6, 8-10, 12-14) and the points disposed far from the field center (1-3, 7, 11, 15-17). The scenarios can be seen in Fig. 5, represented by the red circles, while the blue circles represent the cameras' positions, and the black circles the LiDARs' positions. The centralized scenarios will handle how the LiDAR system deals with equidistant points between both lasers. The dispersed scenarios will be of use to observe how the system deals with points near one laser than the other and in scenarios near the field boundaries.

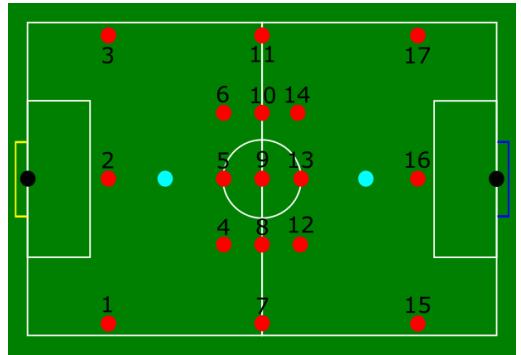


Fig. 5: Positions of the tests.

Besides that, since two cameras are placed for the vision system, the scenarios will be divided into three sets. That will help to see the differences between the captures made by each and observe how they work in the overlap region located along the middle of the field. The sets will be called Quadrant. The first deals with scenarios 1-6, since only the left camera captures these scenarios, and the second deals with scenarios 7-14, which are the scenarios positioned on the overlap of both cameras, and the third deals with scenarios 15-17, the scenarios captured only by the right camera. These sets will show how the SSL-Vision deals with distortion caused by the fish eye effect, mainly on the borders of the image captured by them. It's supposed a substantial error in the output coordinates of the SSL-Vision when they are far from the camera's center.

#### V. RESULTS

Tables I and II summarize the results achieved by the laser and vision for the X and Y axes, respectively, in the 17 scenarios. The Real column indicates the known coordinates of the robot's position, and the Laser and Vision columns show the mean and standard deviation of the samples measured for

TABLE I: Results achieved for the X axis in the SSL field.

Scenario	Real [cm]	Laser [cm]	Vision [cm]
1	-91,5	-91.797 ± 0.584	-92.945 ± 0.026
2	-88,1	-87.594 ± 0.437	-89.099 ± 0.023
3	-93	-92.510 ± 0.391	-93.397 ± 0.049
4	-55	-54.584 ± 0.703	-55.389 ± 0.037
5	-51	-50.321 ± 0.655	-51.323 ± 0.017
6	-54,5	-54.504 ± 0.926	-55.363 ± 0.037
7	0	1.092 ± 0.674	0.488 ± 0.012
8	0	0.390 ± 0.624	0.370 ± 0.073
9	0	0.552 ± 0.513	-0.158 ± 0.061
10	0	0.786 ± 0.478	-0.505 ± 0.036
11	0	0.972 ± 0.953	0.178 ± 0.025
12	61,3	61.859 ± 0.649	61.588 ± 0.017
13	52	51.964 ± 0.519	52.379 ± 0.011
14	53	53.742 ± 1.036	52.868 ± 0.046
15	101	101.653 ± 1.450	100.866 ± 0.015
16	99	99.683 ± 0.554	98.785 ± 0.017
17	97	97.526 ± 1.509	96.679 ± 0.018

TABLE II: Results achieved for the Y axis in the SSL field.

Scenario	Real [cm]	Laser [cm]	Vision [cm]
1	-138,5	-137.784 ± 0.726	-139.464 ± 0.047
2	0	0.844 ± 1.475	0.365 ± 0.018
3	132	131.688 ± 0.625	134.255 ± 0.022
4	-85	-84.104 ± 0.851	-85.172 ± 0.016
5	0	-0.518 ± 1.424	0.690 ± 0.036
6	81,5	80.801 ± 1.435	83.025 ± 0.018
7	-141	-140.488 ± 0.742	-139.829 ± 0.014
8	-82,5	-82.172 ± 0.978	-82.306 ± 0.061
9	0	0.668 ± 1.052	0.963 ± 0.041
10	83,5	84.431 ± 1.781	85.154 ± 0.028
11	125,5	125.017 ± 0.925	126.895 ± 0.083
12	-92,3	-92.848 ± 0.900	-91.665 ± 0.015
13	0	-0.568 ± 0.879	0.810 ± 0.014
14	85,3	85.046 ± 1.317	86.171 ± 0.053
15	-135,5	-135.906 ± 0.813	-135.158 ± 0.022
16	0	0.048 ± 0.598	1.236 ± 0.016
17	123,5	123.125 ± 1.015	124.690 ± 0.023

each system. All the measurements in these and the following tables are in centimeters.

Table III shows the mean error and standard deviation of all the scenarios, the centralized and dispersed sets of points, in the X and Y axes. It is possible to analyze that the mean error for the X-axis of both systems is comparable, with a difference of almost 0.1cm between them. For the Y axis in all scenarios, it's possible to see a considerable difference between them, near 0.43cm. This difference can be explained by the distortion occurring in the borders of the cameras' image, increasing the error in these points since they have a distance greater from the cameras on the Y-axis than the X-axis.

TABLE III: Scenarios mean error and standard deviation.

	X - Laser	X - Vision	Y - Laser	Y - Vision
Mean error	0,5519	0,4461	0,5356	0,9665
Error - Centralized	0,4626	0,3785	0,6011	0,8348
Error - Dispersed	0,6523	0,5221	0,4620	1,1147
Mean Std. Dev.	0,7444	0,0305	1,0315	0,0310
Std. Dev. - Centralized	0,6781	0,0372	1,1796	0,0313
Std. Dev. - Dispersed	0,8190	0,0231	0,8648	0,0306

The errors between the centralized and dispersed sets show that the camera system deals better with the positions in the center of the field, which the distortion on the borders of the cameras' image can explain. That effect cannot be seen when

analyzing the LiDAR system, showing that the detection made by it is regular in all parts of the fields, and also proving the advantages of using two LiDARs to enhance the detection of the circular pattern in the field.

Analyzing the standard deviation metric in Table III, it is possible to notice that the camera system of the SSL is much better than the LiDARs system since the detection was almost 0,3mm dispersed from the mean when observing all scenarios in X, and less than 0,4mm and 0,3mm for the centralized and dispersed scenarios, respectively. For the Y-axis, the results were practically the same. Compared with the LiDAR system, this was worse, but the results are still acceptable, and the laser points image resolution used to perform the pattern detection can explain it. The laser points image was 1000x1000 pixels, each of them means 0.533cm in the real world because of the scale. Thus, the minor difference in the detection leads to a half-centimeter difference in the robot position.

Analyzing the quadrants, Tables IV and V show the results summarized for them in the X and Y axes. Notably, the quadrant of the left camera is worse for the X-axis when analyzing the mean error, with more than double the error when compared with quadrants 2 and 3. Using a fish eye lens in the left camera can explain this effect since it leads to considerable distortion compared to the right camera. For the Y-axis in all quadrants, it is possible to see that the mean error is worse compared with the LiDAR system at least 50%.

TABLE IV: Results for the quadrants in X axis.

	Error X - Laser	Error X - Vis.	Std. Dev. X - Laser	Std. Dev. X - Vis.
Quad. 1	0,3986	0,7360	0,6224	0,0315
Quad. 2	0,6411	0,3122	0,6807	0,0351
Quad. 3	0,6206	0,2233	1,1710	0,01667

TABLE V: Results for the quadrants in Y axis.

	Error Y - Laser	Error Y - Vis.	Std. Dev. Y - Laser	Std. Dev. Y - Vis.
Quad. 1	0,6641	0,9951	1,0893	0,0261
Quad. 2	0,5844	1,0754	1,0956	0,0454
Quad. 3	0,3665	0,8473	0,9203	0,0238

Analyzing the standard deviation, it's not noticeable any influence of the camera's positioning on both axes. Besides, the camera system is still better than the LiDAR system in all quadrants for both axes. When evaluating the LiDARs system for both axes, it is not possible to notice a significant difference between the quadrants, showing that the detection made by the lasers is regular, and the errors accumulated in long distances for one laser are compensated by the other laser.

## VI. CONCLUSION

This work presents an approach for position ground truth using two LiDARs, especially for round robots. The system is compared with the camera system of the Small Size League from RoboCup. Ground truth is needed when developing a localization system since evaluating the estimated position with the robot's actual localization in the real world is crucial. Besides, reliable ground truth works alongside a localization system for calibrating it, like a sensor fusion for indoor environments, odometry adjustment, or camera calibration.

According to the results presented previously in Section V, the developed ground truth system had good results in the mean error calculated between the system output and the accurate coordinates measured using a measuring tape. In both axes, the mean error achieved for the developed system was almost 0.5cm, which is better than the camera system on the Y axis (an error of approximately 0.96cm) and about 0.1cm worse on the X axis. These comparisons are also noticeable when analyzing the centralized and dispersed scenarios.

When analyzing the standard deviation, it is possible to notice that the camera system is much better than the LiDARs system since the standard deviation of the second one is 0.74cm and 1.03cm for the X and Y axes, respectively, while for the vision system, it is near 0.03cm for both axes. These insights are noticeable when analyzing the centralized and dispersed scenarios. The scale and image resolution (1000x1000 pixels) used for the LiDAR system can explain these results.

When analyzing the quadrants described, it is possible to confirm that the vision system was worse when comparing the LiDAR system on the Y-axis. For the X-axis, it is noticeable that one of the cameras used distorted the correct position of the robot because of the use of a lens in it. The standard deviation of each quadrant followed the insights observed in the mean of all scenarios, which is the vision system being much superior compared with the laser system.

In conclusion, the LiDARs system developed can be used as a position ground truth system for round robots since the mean error and mean standard deviation results presented are not notable, practically less than 1 centimeter on both. Besides, the results could be enhanced using more powerful LiDARs that provide more points and better precision over the distance. Besides, the LiDAR scan frequency of 10Hz is a bottleneck since it limits the system detection frequency. The detection can occur for any diameter, depending on the scale used in positioning the laser data in the laser points image.

Future works could include adding a Kalman filter to have a smooth system and decrease the possible variations in the detection. Besides, using a more powerful notebook aims to increase the LiDAR system image resolution, which should provide a better standard deviation to the system. Moreover, adding at least one more LiDAR can increase the system's accuracy, thus achieving results even closer to the ground truth, and it can be possible to determine the robot's orientation by detecting the robot's dribbler system.

#### ACKNOWLEDGEMENTS

The authors would like to thank the University Center of FEI, the RoboFEI team and CAPES for their support.

#### REFERENCES

- [1] Aoki, S., Degawa, T., Fujihara, K., Notsu, Y., Beppu, T.: Mct susano logics 2016 team description. Available for download in [http://wiki.robocup.org/images/2/27/Small Size League-RoboCup \(2016\)](http://wiki.robocup.org/images/2/27/Small%20Size%20League-RoboCup%20(2016).pdf)
- [2] Beppu, T., Aoki, S., Horiuchi, T.: A shared multi-particle filter for ball position estimation in robocup small size league soccer games. In: 2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC). pp. 311–316 (2016). <https://doi.org/10.1109/ICARSC.2016.7477002>
- [3] Chen, W., Xu, J., Zhao, X., Liu, Y., Yang, J.: Separated sonar localization system for indoor robot navigation. *IEEE Transactions on Industrial Electronics* **68**(7), 6042–6052 (2021). <https://doi.org/10.1109/TIE.2020.2994856>
- [4] English, C., Zhu, S., Smith, C., Ruel, S., Christie, I.: Tridar: A hybrid sensor for exploiting the complimentary nature of triangulation and lidar technologies. In: Proceedings of the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space. vol. 1 (2005)
- [5] Guan, W., Chen, S., Wen, S., Tan, Z., Song, H., Hou, W.: High-accuracy robot indoor localization scheme based on robot operating system using visible light positioning. *IEEE Photonics Journal* **12**(2), 1–16 (2020). <https://doi.org/10.1109/JPHOT.2020.2981485>
- [6] Hassanein, A.S., Mohammad, S., Sameer, M., Ragab, M.E.: A survey on hough transform, theory, techniques and applications. arXiv preprint arXiv:1502.02160 (2015)
- [7] Huang, Z., Chen, L., Li, J., Wang, Y., Chen, Z., Wen, L., Gu, J., Hu, P., Xiong, R.: Zjuniject extended team description paper for robocup 2019. RoboCup (2019)
- [8] Khandelwal, P., Stone, P.: A low cost ground truth detection system for robocup using the kinect. In: RoboCup 2011: Robot Soccer World Cup XV 15. pp. 515–527. Springer (2012)
- [9] Logitech: BRIO ULTRA HD BUSINESS WEBCAM (3 2021)
- [10] Lotufo, R.A., Audigier, R., Saúde, A.V., Machado, R.C.: Morphological image processing. In: Microscope image processing, pp. 75–117. Elsevier (2023)
- [11] Marchant, R., Guerrero, P., Ruiz-del Solar, J.: A portable ground-truth system based on a laser sensor. In: RoboCup 2011: Robot Soccer World Cup XV 15. pp. 234–245. Springer (2012)
- [12] Martins, A., Dias, A., Silva, H., Almeida, J., Gonçalves, P., Lopes, F., Faria, A., Ribeiro, J., Silva, E.: Groundtruth system for underwater benchmarking. In: 2013 OCEANS - San Diego. pp. 1–5 (2013). <https://doi.org/10.23919/OCEANS.2013.6741307>
- [13] Mukhopadhyay, P., Chaudhuri, B.B.: A survey of hough transform. *Pattern Recognition* **48**(3), 993–1010 (2015)
- [14] Niemüller, T., Ferrein, A., Eckel, G., Pirro, D., Podbregar, P., Kellner, T., Rath, C., Steinbauer, G.: Providing ground-truth data for the nao robot platform. In: RoboCup 2010: Robot Soccer World Cup XIV 14. pp. 133–144. Springer (2011)
- [15] Nieto, J., Bailey, T., Nebot, E.: Recursive scan-matching slam. *Robotics and Autonomous Systems* **55**(1), 39–49 (2007). <https://doi.org/https://doi.org/10.1016/j.robot.2006.06.008>, <https://www.sciencedirect.com/science/article/pii/S0921889006001461>, simultaneous Localisation and Map Building
- [16] Panigrahi, P.K., Bisoy, S.K.: Localization strategies for autonomous mobile robots: A review. *Journal of King Saud University - Computer and Information Sciences* **34**(8, Part B), 6019–6039 (2022). <https://doi.org/https://doi.org/10.1016/j.jksuci.2021.02.015>, <https://www.sciencedirect.com/science/article/pii/S1319157821000550>
- [17] Pennisi, A., Bloisi, D.D., Iocchi, L., Nardi, D.: Ground truth acquisition of humanoid soccer robot behaviour. In: Behnke, S., Veloso, M., Visser, A., Xiong, R. (eds.) RoboCup 2013: Robot World Cup XVII. pp. 560–567. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
- [18] RoboCup: Robocup objective (9 2019), <https://ssl.robocup.org/rules/>, access in 23 dec. 2024 at 21:33
- [19] RoboCup-SSL: Small size league rules (5 2024), <https://ssl.robocup.org/rules/>, access in 23 dec. 2024 at 21:33
- [20] SLAMTEC: RPLIDAR A1 - Low Cost Degree Laser Range Scanner (7 2016), rev. 10
- [21] SLAMTEC: Slamtec lidar ros2 package. [https://github.com/Slamtec/rplidar\\_ros/tree/ros2](https://github.com/Slamtec/rplidar_ros/tree/ros2) (2016)
- [22] SLAMTEC: RPLIDAR S1 - Low Cost Degree Laser Range Scanner (1 2018), rev. 10
- [23] Sorokumov, S., Glazunov, S., Chaika, K.: Distributed visual-based ground truth system for mobile robotics
- [24] Technologies, A.V.: Technical Manual (4 2011), v4.4.2
- [25] Yang, T., Li, Y., Zhao, C., Yao, D., Chen, G., Sun, L., Krajinik, T., Yan, Z.: 3d tof lidar in mobile robotics: A review. arXiv preprint arXiv:2202.11025 (2022)
- [26] Zickler, S., Laue, T., Birbach, O., Wongphati, M., Veloso, M.: Ssl-vision: The shared vision system for the robocup small size league. In: Baltes, J., Lagoudakis, M.G., Naruse, T., Ghidary, S.S. (eds.) RoboCup 2009: Robot Soccer World Cup XIII. pp. 425–436. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)