RoboCIn - Equipe de Robótica do Centro de Informática

Carlos Henrique Caloete Pena, Cristiano Santos de Oliveira, Gabriel Marques Bandeira,
Heitor Rapela Medeiros, Iraline Nunes Simões, Jose Nilton de Oliveira Lima Júnior,
Lucas Henrique Cavalcanti Santos, Marvson Allan Pontes de Assis,
Renato Sousa Bezerra, Roberto Costa Fernandes, Vinicius Bezerra Araújo da Silva,
Hansenclever de Franca Bassani, Edna Natividade da Silva Barros
Centro de Informática
Universidade Federal de Pernambuco
Pernambuco, Brasil
Email: {chcp, cso, gmb, hrm, ins, jnolj, lhcs, mapa, rsb5, rcf6, vbas, hfb, ensb}@cin.ufpe.br

***Abstract*—Este artigo tem como objetivo descrever o projeto desenvolvido pela equipe RoboCIn, do Centro de Informática da UFPE para participação na competição IEEE Very Small Size Soccer. Neste artigo são descritos os principais sistemas desenvolvidos: o sistema de localização por visão computacional, a técnica desenvolvida para planejamento de caminho baseada em campos potenciais, bem como o projeto eletrônico e mecânico do robô e o chassis projetado para a categoria citada.**

I. INTRODUÇÃO

Para um projeto da categoria IEEE Very Small Size Soccer, é necessário quatro pilares: detecção de objetos, planejamento de caminho e ações, mecânica e eletrônica. A primeira etapa do ciclo de execução do projeto é a etapa de detecção de objetos, nesta etapa, a equipe optou por criar um módulo de detecção da bola e dos robôs utilizando visão computacional. Após a etapa de detecção, as informações de posição, direção e velocidade são passadas para o módulo de planejamento. O módulo de planejamento define um caminho a ser seguido pelos robôs e envia as velocidades para o módulo de controle do robô (*hardware* embutido no robô). Com as velocidades definidas, o módulo de controle local do robô realiza o controle do robô. Tal robô, por sua vez, foi criado pelo pilar da mecânica. Na próxima seção, é explicado o módulo de localização dos objetos em campo. Na seção seguinte, é detalhado o planejamento de caminho e ações realizado pelo *software* da equipe. Em seguida as seções de mecânica e eletrônica dos robôs criados e finalizando com a seção da comunicação entre os módulos e a conclusão e agradecimentos, além das referências.

II. LOCALIZAÇÃO

Para identificação e detecção dos robôs e bola em campo, foi desenvolvido pela nossa equipe um software de Visão Computacional em C++, utilizando a biblioteca open-source OpenCV [1] baseado em detecção de cores de etiquetas, localizadas na parte superior de cada robô. Para captura das imagens necessárias para detecção, é utilizada uma webcam posicionada a 2 metros do campo. O software de visão computacional pode ser visto como um software de arquitetura de Pipes e Filtros [2] com as seguintes etapas: Pré-processamento, Segmentação, Identificação de objetos e Transformação de sistema de coordenadas.

A. PRÉ-PROCESSAMENTO

Na etapa de pré-processamento são utilizados primariamente configurações internas da própria câmera, como controle de brilho e ajuste de foco automático. Tais configurações são definidas via *bash* através de um *script* gerado pela equipe e apresenta uma grande vantagem em termos de velocidade, uma vez que tais configurações são definidas internamente na câmera, em nível de *hardware*. Além disso, é feito a cada novo *frame* uma análise e equalização cumulativa de histogramas, procurando dessa forma eliminar grandes atenuações nas sequências de *frames*, que podem ser causadas por um mal funcionamento momentâneo do *hardware*.

B. SEGMENTAÇÃO

Na etapa de segmentação é feita a separação das imagens em informações úteis para as próximas etapas, eliminando partes das imagens que não são necessárias para a identificação dos objetos. Foi escolhido o desenvolvimento de uma segmentação de cores com a utilização de uma Look Up Table (LUT) [3]. A segmentação de cores se baseia na análise das intensidades dos canais de cor de cada pixel, onde o pixel se configurará como um pixel de interesse se a intensidade dos canais de cor do pixel está presente dentro de um dos intervalos da cor característica de alguma etiqueta ou bola. A utilização de uma LUT aparece como uma complementação da técnica de segmentação através de cores, tornando o processo de segmentação mais veloz, uma vez que elimina a necessidade de passar várias vezes por todos os pixels da imagem, sendo uma vez para cada cor. Para garantir mais confiabilidade do processo de segmentação, os valores guardados na LUT são de intensidades no espaço YUV,

espaço que garante mais confiabilidade pela sua característica da luminosidade está presente em somente 2 dos 3 canais de intensidade, tornando assim a segmentação mais robusta a mudanças de luminosidade se comparado ao espaço RGB ou HSV. Assim, para se fazer a consulta na LUT, é necessário fazer a conversão de BGR (espaço utilizado pela câmera na captura de frames) para YUV.

Após o processo de segmentação, será passado para o módulo seguinte uma matriz com os elementos de interesse, como pode ser visto na Figura 1.

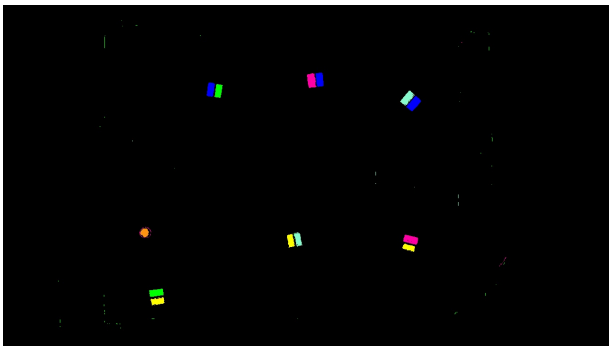

Fig. 1. Elementos de interesse identificados.

C. IDENTIFICAÇÃO DE OBJETOS

Na etapa de reconhecimento foi utilizada variações do algoritmo de clusterização k-means [4], onde é agrupado os robôs de mesmo time através das cores principais.

Primeiramente, são posicionados N *clusters* de forma aleatória, 2 para cada robô e um para a bola. Tal processo é apresentado na Figura 2. Cada *cluster* tem sua cor de referência e este só irá se incorporar com pixels da mesma cor. Então cada pixel tem sua distância comparada a todos os *clusters* que contenha a mesma cor e será incorporado ao centróide que estiver a uma menor distância euclidiana, como pode ser visto na Figura 3.

Após a varredura da matriz, o centróide irá para o ponto que contém a média das coordenadas entre os pontos associados. O processo será repetido até que gere uma convergência dos núcleos, e a posteriori para cada robô é unido 2 núcleos, um representando a cor do time, amarelo ou azul, e a outra a cor escolhida pelo time (cor secundária). No próximo momento o algoritmo é iniciado novamente, entretanto ao invés dos núcleos serem iniciados aleatoriamente é escolhida a posição do *frame* anterior.

D. CONVERSÃO DE COORDENADAS

Por fim, foi necessário fazer a conversão dos pontos adquiridos no plano da imagem para pontos em escala real. Para alcançar o resultado desejado, utilizamos métodos de interpolação de coordenadas, utilizando correspondências já conhecidas de pontos da imagem com os pontos reais para gerar uma transformada que execute a conversão desejada.

Fig. 2.

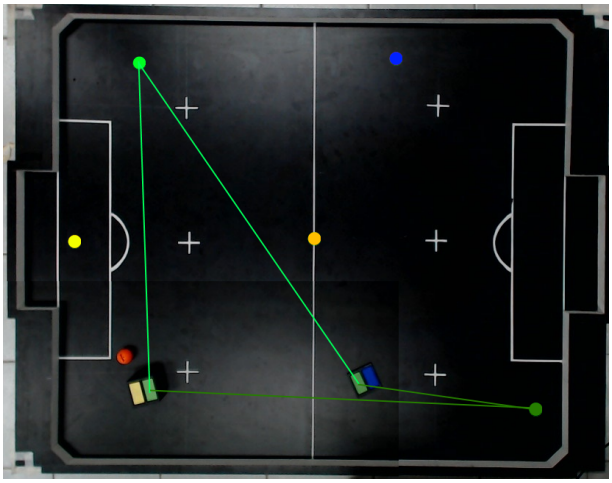

Fig. 3.

III. PLANEJAMENTO

A estratégia tem como principal objetivo definir as posições para as quais os robôs tem que se locomover. A partir das coordenadas de cada elemento no campo de jogo (bola e robôs), a estratégia utiliza o algoritmo de campos potenciais [7] para definir uma posição alvo para o robô e o caminho até esta. Estes dados são convertidos em velocidades e sentido de rotação dos motores responsáveis por conduzir as rodas dos robôs.

Utilizou-se o software de simulação VREP para a realização de testes do algoritmo, visto que foi possível desenvolver um ambiente similar ao real e, assim, modular diversos cenários de jogo sem haver uma dependência técnica das outras áreas da equipe (visão e robô). Uma imagem da simulação pode ser vista na figura 4.

O algoritmo de campos potenciais é uma estratégia muito utilizada para planejamento de rota de robôs. Neste, o robô é um ponto que sofre a influência de um campo potencial gerado pelos outros robôs, pela bola, pelo seu

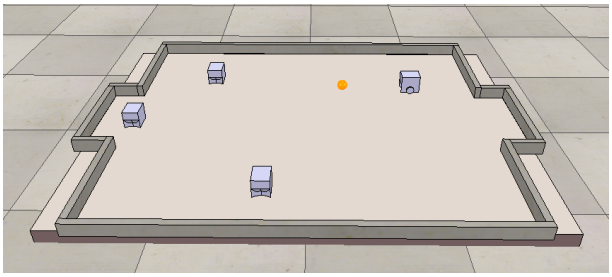

Fig. 4. Exemplo de situação de jogo simulada no VREP.

objetivo final e pelo campo de jogo, onde as posições a serem evitadas são pontos com maiores potenciais e as posições mais próxima do objetivo do robô são pontos com menores potenciais. Para definir o alcance do campo, de cada obstáculo e do objetivo, foram geradas gaussianas (positivas ou negativas) que utilizam a equação 1, onde os parâmetros σ_x e σ_y definem o alcance da Gaussiana, A é a amplitude máxima ou mínima da Gaussiana, x_o e y_o são as coordenadas do obstáculo, ou do objetivo, e x e y são as coordenadas do ponto que está sendo analisado. Um exemplo de campos potenciais gerados para a figura 4 é a figura 5.

$$f(x,y) = Ae^{-(\frac{(x-x_o)^2}{2\sigma_x} + \frac{(y-y_o)^2}{2\sigma_y})} \quad (1)$$

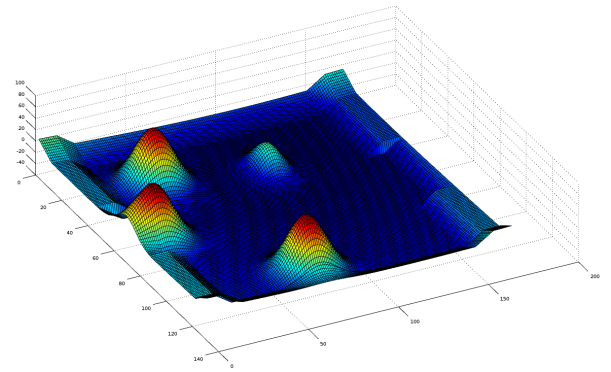

Fig. 5. Campos potenciais gerados para simulação.

Um dos problemas encontrados com o uso de campos potenciais foi que em algumas situações são gerados mínimos locais indevidos. Nestas posições, o robô tende a ficar estagnado, pois o algoritmo de planejamento de caminhos tende a achar que esse local é o objetivo final, apesar de não o ser. Para tratar esse problema foi utilizado uma função de custo de busca em grafos, que tende buscar caminhos com menor custo. Para o cálculo do custo de um dado ponto leva-se em conta o custo para se chegar a esta posição, o valor do campo potencial desta posição e a distância do ponto até o objetivo. Esse algoritmo computa os custos de cada rota de navegação antes de enviar as informações ao robô. Com esse cálculo, caminhos com mínimos locais ficam com um custo muito alto o que tornará este caminho indevido, e leva o algoritmo a buscar caminhos com custos menores, que levam ao caminho certo.

Para a navegação acontecer da maneira que se espera foi implementada uma função que calcula a velocidade de cada roda e corrige pequenas alterações no caminho, que é um problema recorrente no âmbito da robótica. Na maioria das vezes, os valores calculados para que o robô chegue ao objetivo não são os mesmos que o *hardware* executa. Isso acontece devido a pequenas variações de um motor para o outro como algumas outras calibrações necessárias dentro do *hardware*. Então essa função compara via *software* o movimento executado pelo robô com o que é esperado e com isso faz as correções em tempo de execução. Para exemplificar, uma rota em linha reta sem essas correções pode se tornar um curva leve causada por um erro de calibração em algum dos motores ou deslizamento das rodas.

IV. MECÂNICA

O robô foi projetado com o intuito de ser modular, e de fácil montagem e manutenção. Portanto toda a estrutura para construção do robô foi projetada para ser construída através de impressão 3D, como pode-se observar na 6. Os módulos da estrutura são encaixáveis, os quais não necessitam de parafusos para estruturar a montagem do robô. Para projetar todo o robô e seus encaixes, utilizou-se o Solid Works 3D CAD [5], o qual permite completa modelagem e exportação de peças para impressão 3D.

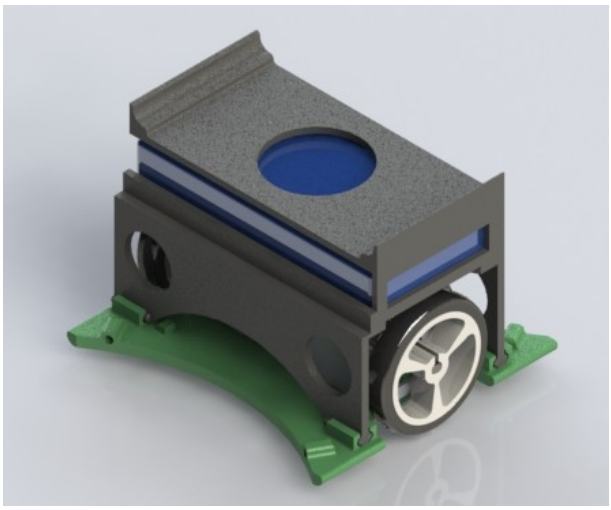

Fig. 6. Estrutura interna do robô e seus módulos estruturais.

Aliado a uma estrutura impressa em 3D, o robô da equipe RoboCIn possui dois micro motores com redução integrada de 75:1, e permitem o eixo rodar a 400 RPM com torque de 1,6 Kg-cm. Para completar a estrutura de locomoção do robô existem duas rodas com 32mm de diâmetro, localizadas nas laterais do robô, e 4 pivôs já inclusos na estrutura impressa.

V. ELETRÔNICA

Para controlar o robô é utilizado uma Arduino Pro Mini de 16 MHz, escolhida pela praticidade, pequeno tamanho, e capacidade suficiente para controlar todos os dispositivos requisitados no robô. Para controle de potência nos motores é utilizado o driver de motor TB6612FNG, o qual possui limite de corrente contínua de 1A por motor e 3A para corrente de pico.

No circuito do robô existe também um módulo xBee, responsável pela comunicação e uma IMU (Inertial Measurement Unit), a qual permite monitorar a orientação do robô, assim como as velocidades de rotação. E para praticidade foi incluído no circuito algumas chaves, as quais permitem alterar o ID dos robôs sem necessitar alterar o *software* do controlador.

Para otimizar o espaço do robô, o circuito foi impresso em placas de dupla camada através da máquina LPKF. Para realizar a produção do circuito impresso foi utilizado o Eagle [6] (ver figura 7), software especializado para produzir placas de circuito impresso.

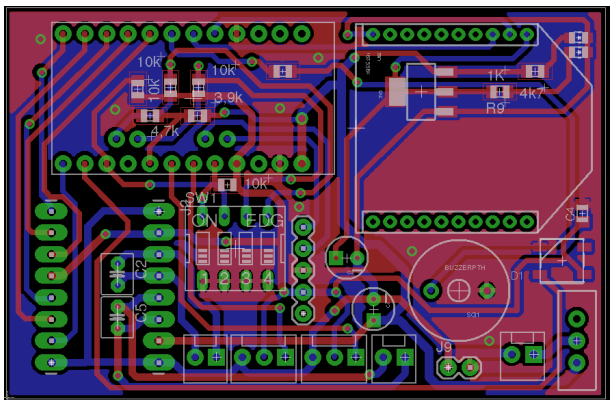

Fig. 7. Projeto do circuito feito através do Eagle Circuit CAD.

VI. COMUNICAÇÃO

Na parte de comunicação, foi escolhida a tecnologia ZigBee para a troca de mensagens entre o computador e os robôs. Com esta tecnologia, foi possível que o computador enviasse informações como velocidade dos motores e receber dados como o nível de bateria de cada robô. Para realizar a comunicação, o computador envia uma mensagem *broadcast* contendo o ID do robô de destino, a velocidade de cada motor e se é para o robô retornar o nível de bateria. A Figura 8 mostra como foram divididos os bits da comunicação para conter as informações citadas.

VII. CONCLUSÃO

No presente *Team Description Paper* (TDP), é mostrado como pode ser feito um sistema completo para a participação de uma equipe na IEEE *Very Small Soccer League*. Para isso, foi descrito um sistema de visão computacional para detectar robôs com um padrão de cores na face superior, um planejador de rotas e decisões a fim

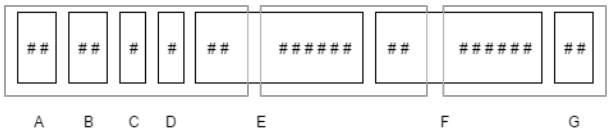

Fig. 8. Protocolo de comunicação. A: Início de comunicação. B: Identificador do robô de destino. C: Bit indicativo de retorno de nível de bateria. D: Bit indicativo de chute (não utilizado nesta versão). E: Velocidade do motor 1. F: Velocidade do motor 2. G: Fim de comunicação..

de que os jogadores do time tentem vencer a partida e o esquema do que é necessário para criar os robôs de tal sistema.

AGRADECIMENTOS

A equipe gostaria de agradecer o Centro de Informática da UFPE pelo apoio financeiro e de recursos durante todo o processo do projeto. Também gostaríamos de agradecer à todo apoio dado pelos professores Edna Barros e Hansenclever Bassani ao longo da criação do projeto.

References

[1] OpenCV. Disponível em: http://opencv.org/. Acessado por último em 20 de junho de 2016.
[2] Sommerville, Ian. Engenharia de Software, 9ª edição. Pearson Education.
[3] Discrete YUV Look-Up Tables for Fast Colour Segmentation for Robotic Applications
[4] Hartigan, J., & Wong, M. (1979). Algorithm AS 136: A K-Means Clustering Algorithm. Journal of the Royal Statistical Society. Series C (Applied Statistics), 28(1), 100-108. doi:1. Retrieved from http://www.jstor.org/stable/2346830
[5] CAD Solid Works. Disponível em: http://www.solidworks.com/. Acessado por último em 20 de junho de 2016.
[6] CAD Eagle. Disponível em: http://www.cadsoftusa.com/. Acessado por último em 20 de junho de 2016.
[7] Roland Siegwart and Illah Reza Nourbakhsh and Davide Scaramuzza Introduction to Autonomous Mobile Robots. The MIT Press, 2ª edição 2011.