

PS 2023 - Especificação projeto de Software SSL

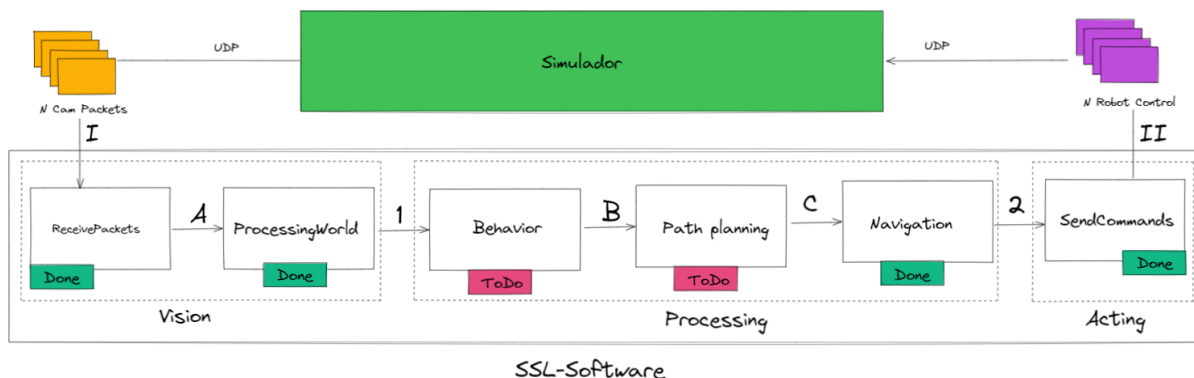
Descrição do problema

No RobôCIn, um dos principais desafios do desenvolvimento de qualquer software é o domínio do fluxo de dados e de processamento ao longo dos módulos código. Para os softwares das categorias VSS (*Very Small Size Soccer*) e SSL (*Small Size League*), outro desafio ainda maior é analisar o contexto do ambiente atual da partida em conjunto com as particularidades de como os robôs de cada categoria atua no ambiente e das regras, em como definir qual é a melhor ação a ser tomada agora por cada jogador disponível.

Projeto da Seletiva

Pensando nisso, para o projeto de *software* da seletiva 2022, será disponibilizada uma base comum de *software* que representa o ciclo básico de decisão nas categorias de futebol de robôs. O principal desafio deste projeto é entender a estrutura de funcionamento do software para propor/desenvolver melhorias e aprimoramentos de alguns dos módulos disponibilizados.

A Figura a seguir resume o fluxo de funcionamento do *software* do RobôCIn:



- I e II, representam a comunicação entre o *software* e o simulador, recebendo e enviando dados respectivamente.
- A, B e C, representam o fluxo que os dados percorrem entre as funções de um módulo.
- 1 e 2, representam a comunicação entre módulos do *software*.

As funções indicadas como *ToDo* na figura são justamente as do processamento, o módulo de mais abertura a avanços, ideias novas e experimentação, o qual exige um entendimento do contexto do futebol de robôs em cada categoria. A principal atividade do projeto a ser desenvolvida é analisar as informações vindas da visão e gerar ações a serem executadas pelos robôs no simulador.

Como sistema operacional, é necessário o linux (Ubuntu ou PopOs!) >= 20.04 (a compilação do projeto exige g++ >= 9.3), ou distribuições equivalentes. Sugerimos como IDE o uso do VSCode. O simulador utilizado será o [GrSim](#), por possuir interface

gráfica que permite fácil visualização e modificação dos posicionamentos de robôs e bola para testes, o mesmo simulador utilizado na LARC (*Latin American Robotics Competition*) nos últimos anos.



Exemplos de *behaviors* que podem ser implementados, qualquer combinação deles é válida, assim como desenvolver outros comportamentos, sinta-se livre para expandir as ideias e trazerem novas

- Pegar a bola e chutar no gol do outro time;
- Pegar a bola, segurar e se mover com ela;
- Dá um passe para outro jogador;
- Roubar a bola de um jogador inimigo;
- Ir até um ponto evitando os obstáculos e colisões;
- Proteger o nosso gol.

Dica 1: Para melhor entendimento do funcionamento de uma partida das categorias, recomendamos procurar vídeos de jogos em competições passadas, além de olhar as regras.

Dica 2: Normalmente, a cada momento, cada ação macro está definida para um número restrito de jogadores, em uma jogada você pode pensar individualmente cada jogador ou coordenar mais de um jogador para atuarem em equipe.

Dica 3: Dependendo das posições, um funcionamento mínimo pode ser obtido apenas implementando behaviors, mas com path planning os behaviors se tornam robustos para um cenário genérico.

Formato da Entrega

- *Link* para um repositório no GitHub com suas implementações

Na sua apresentação, tente responder às seguintes perguntas:

1. Na sua opinião, e após sua pesquisa durante o desenvolvimento do projeto, quais são os maiores desafios na hora de fazer este projeto?
2. Como foi a curva de aprendizado para absorver e compreender o fluxo estabelecido pelo código base?
3. Por que nas competições de Robótica damos preferências a linguagens como C e C++?
4. Se você fosse definir um projeto que tivesse como restrições ser de tempo real, seria uma boa abordagem usar *threads* para executar os behaviors?
5. Caso a resposta anterior for sim, explique como funcionaria a troca de informações entre câmera (Visão), estratégica (*Processing*), Acting (Comunicação) na sua arquitetura. Se possível, construa um diagrama utilizando o draw.io para explicar sua proposta.
6. Você já trabalhou com *classes*? Se sim, seria prática utilizar *classes* para estruturar o desenvolvimento de *software*?

Dica: Procure as regras da competição, e caso tenha dúvidas, pode mandar perguntas para a equipe.

Restrições do projeto

- No RobôCIn, as equipes de software das categorias SSL/VSS utilizam primordialmente a linguagem C/C++.
- O projeto deve conter comentários para ajudar o entendimento por membros da equipe RobôCIn e o mesmo deve ser enviado utilizando a ferramenta [GitHub](https://github.com).

Links Úteis

- Site oficial da categoria SSL - [aqui](#)
- Regras da categoria de SSL - [aqui](#)
- Gravação dos Jogos de SSL da LARC 2021 - [aqui](#)
- Gravação dos Jogos de campeonatos simulados - [aqui](#)
- Site com alguns TDPs (*Team Description Paper*) de SSL - [aqui](#)
- Software base para o desenvolvimento - [aqui](#)
- Simulador GrSim para instalar e executar o simulador. - [aqui](#)
- Projeto base disponibilizado - [aqui](#)

Ficou com alguma dúvida que não foi respondida? Entra em contato com a gente!

- Instagram: <https://www.instagram.com/robocinufpe/>
- Gmail: robocin@cin.ufpe.br