PGCOMP - Programa de Pós-Graduação em Ciência da Computação
Universidade Federal da Bahia (UFBA)
Av. Milton Santos, s/n - Ondina
Salvador, BA, Brasil, 40170-110

https://pgcomp.ufba.br
pgcomp@ufba.br

Um dos grandes desafios em Sistemas Multiagentes (SMA) é a criação de planos cooperativos para lidar com os diversos cenários que se apresentam num ambiente dinâmico, de tempo real, composto por times de robôs móveis. Neste cenário, cada robô é controlado por um agente do (SMA), o qual precisa tomar decisões complexas em um curto espaço de tempo de forma coordenada com os demais robôs de seu time. Apesar das muitas soluções desenvolvidas com base em planejamento multiagente e aprendizagem por reforço, um espectador humano usualmente percebe oportunidades para melhores planos cooperativos em muitos cenários em que os robôs apresentam desempenho abaixo do esperado. A pesquisa apresentada nesta tese consiste em capturar o conhecimento do observador humano para demonstrar como times de robôs podem cooperar melhor na solução do problema que devem resolver. Como consequência, as diversas demonstrações humanas podem ser reunidas em um conjunto de dados para treinamento dos agentes que controlam os robôs. Para o desenvolvimento desta pesquisa, foi utilizado o ambiente RoboCup 3D Soccer Simulation (3DSSIM) e a coleta das demonstrações humanas foi realizada por meio de um conjunto de ferramentas desenvolvido a partir da adaptação de soluções existentes na comunidade RoboCup, utilizando uma estratégia de crowdsourcing. Além disso, foi utilizado o agrupamento fuzzy para reunir demonstrações que tenham o mesmo significado semântico, mesmo que com pequenas diferenças entre elas. Com os dados organizados, um mecanismo de aprendizagem por reforço foi utilizado para aprender uma política de classificação que permite aos agentes decidirem qual o grupo de jogadas é mais adequado a cada situação que se apresenta no ambiente. Os resultados evidenciam a capacidade de evolução do time de robôs, a partir da aprendizagem da política de seleção das jogadas sugeridas e do seu uso de forma adequada às habilidades de cada robô.`

Palavras-chave: Sistemas Multiagentes; Aprendizagem por Reforço; Agrupamento Fuzzy; RoboCup; Futebol de Robôs.

DSC | 029 | 2022

Aprendizagem por Demonstração de Planos Coordenados em Sistemas Multiagentes

Marco Antonio Costa Simões

UFBA

Aprendizagem por Demonstração de Planos Coordenados em Sistemas Multiagentes

Marco Antonio Costa Simões

Tese de Doutorado

Universidade Federal da Bahia

Programa de Pós-Graduação em
Ciência da Computação

Julho | 2022

Universidade Federal da Bahia
Instituto de Computação

Programa de Pós-Graduação em Ciência da Computação

APRENDIZAGEM POR DEMONSTRAÇÃO DE PLANOS COORDENADOS EM SISTEMAS MULTIAGENTES

Marco Antonio Costa Simões

TESE DE DOUTORADO

Salvador
5 de julho de 2022

MARCO ANTONIO COSTA SIMÕES

APRENDIZAGEM POR DEMONSTRAÇÃO DE PLANOS COORDENADOS EM SISTEMAS MULTIAGENTES

Esta Tese de Doutorado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

Orientadora: Tatiane Nogueira Rios

Salvador
5 de julho de 2022

Ficha catalográfica elaborada pela Biblioteca Universitária de Ciências e Tecnologias Prof. Omar Catunda, SIBI - UFBA.

C871 Simões, Marco A. C.

Aprendizagem por Demonstração de Planos Coordenados em Sistemas Multiagentes/ Marco A. C. Simões. – Salvador, 2022.

123 f.

Orientadora: Profª. Drª Tatiane Nogueira Rios

Tese (Doutorado) – Universidade Federal da Bahia. Instituto de Computação, 2022.

1. Inteligência Artificial. 2. Sistema Multiagentes. 3. Aprendizagem por Demonstração. 4. Agrupamento Fuzzy. I. Rios, Tatiane Nogueira. II. Universidade Federal da Bahia. III. Título.

CDU:616-083:173.4

MARCO ANTONIO COSTA SIMOES

APRENDIZAGEM POR DEMONSTRAÇÃO DE PLANOS COORDENADOS EM SISTEMAS MULTIAGENTES

Esta tese foi julgada adequada à obtenção do título de Doutor em Ciência da Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação da UFBA.

Salvador, 05 de julho de 2022

Profª. Drª. Tatiane Nogueira Rios
(Orientador - PGCOMP/UFBA)

Prof. Dr. Luis Paulo Gonçalves dos Reis
(Universidade do Porto, Portugal)

Prof. Dr. Reinaldo Augusto da Costa Bianchi
(FEI-SP)

Prof. Dr. João Alberto Fabro
(UTFPR)

Profª. Drª. Rita Suzana Pitangueira Maciel
(PGCOMP/UFBA)

À minha esposa Cintia, a meus filhos Bruno e Beatriz e à minha mãe Silvia $(in\ memorian)$.

AGRADECIMENTOS

Uma jornada longa e difícil como esta que se encerra com a produção deste documento, é recheada de apoiadores e também de obstáculos. Aqui registro minha profunda a gratidão a todos, buscando destacar os mais relevantes. Inicialmente, agradeço a Deus, inteligência suprema e causa primária de todas as coisas. Sem a inspiração e permissão da providência divina, nenhuma das oportunidades que pude aproveitar estariam disponíveis, sequer a minha própria existência. Agradeço ainda à espiritualidade superior que me intuiu e me guiou nos momentos mais difíceis desta jornada. Entre os encarnados, agradeço profundamente à minha orientadora Tatiane Nogueira Rios, uma pessoa de profundo conhecimento e sensibilidade, capaz de compreender a situação atípica à qual me submeti para fazer este processo de doutoramento. Tanto sua paciência e compreensão, quanto seu apoio técnico nos momentos decisivos foram fundamentais para o êxito desta empreitada. Agradeço a todo o corpo docente do Programa de Pós-Graduação em Ciência da Computação (PGCOMP) pela qualidade dos ensinamentos e experiências compartilhadas. A Universidade Federal da Bahia é a casa onde conquistei o meu primeiro grau do ensino superior e agora também o derradeiro. Em especial, agradeço aos professores George Marconi Lima e Raimundo Macêdo, meus orientadores na primeira tentativa de doutoramento nesta instituição, que não obteve êxito por questões particulares, mas serviu de embasamento e fundação para o êxito agora conquistado. Não poderia deixar de agradecer à minha amada família, a quem dedico este trabalho. Minha esposa, Cintia, sempre paciente e suprindo minhas ausências nos momentos mais críticos, sendo compreensiva, me apoiando nos momentos mais difíceis. Aos meus filhos, Bruno e Beatriz, pela compreensão nos momentos de ausência e de sacrifícios que foram necessários, especialmente na reta final deste trabalho. Retornando à pátria espiritual, agradeço à minha mãe Silvia (*in memorian*) por todos os exemplos e valores morais que me foram ensinados, pela demonstração de resiliência e resignação para enfrentar as situações mais difíceis em sua jornada aqui na Terra, e por me ensinar o valor que a educação tem na formação moral e intelectual do ser humano. De volta à Terra, agradeço ao meu pai Valmir por sempre ter me proporcionado todas as condições para desenvolver meus estudos nas idades mais tenras nas melhores instituições da nossa cidade, permitindo que agora eu alcance este grau tão valioso. Agradeço também, pai, pelo exemplo de seguir sempre no trabalho intenso e firme sem jamais desistir das conquistas almejadas. Gratidão à minha sogra Adenice pelo suporte fornecido à minha esposa e filhos, nos momentos em que precisei me ausentar. Esta rede de apoio foi essencial. Agradeço aos meus irmãos Ricardo e Marta e minhas afilhadas queridas Fernanda e Maria Clara, por todo o carinho dispensando nos momentos de convivência durante esta jornada. Agradeço profundamente todos os meus alunos, orientandos de IC e TCC, na Universidade do Estado da Bahia(Uneb). Em especial, agradeço àqueles que envolveram-se direta ou indiretamente na execução deste

trabalho: Claudia Elizabete, Matheus, Caroline Souza, Jadson Nobre, Vitor Manoel dos Santos, Felipe Mascarenhas, Rafael Fonsêca e Gabriel Sousa. Agradeço também aos colegas pesquisadores do Centro de Pesquisa em Arquitetura de Computadores, Sistemas Inteligentes e Robótica(ACSO) da Uneb: Ana Patricia, Josemar, Jorge e Robson. Vocês foram fundamentais tanto no apoio durante todo o tempo, quanto ao assumir algumas das minhas atribuições em diversas etapas desta trajetória. Este título conquistado é nosso, é do ACSO. Também deixo a minha gratidão para todos os amigos que fiz durante dezesseis anos participando da comunidade RoboCup, tanto internacional quanto no Brasil. O apoio e conhecimentos adquiridos com vocês durante todos estes anos foram essenciais para o cumprimento dos objetivos propostos neste trabalho. Agradeço também aos colegas do TRT5, especialmente à equipe do Escritório de Segurança da Informação e diretoria da SETIC por todo o apoio recebido. Grandes coisas sempre são resultado do trabalho coletivo e por mais que o doutorado se proponha a ser um trabalho individual, ninguém consegue vencer este desafio sozinho. Por isto deixo minha profunda gratidão a todos os citados e também àqueles que porventura não tenham sido mencionados mas que estão sempre inclusos no meu sentimento de gratidão. Obrigado!

Todo efeito tem uma causa. Todo efeito inteligente tem uma causa inteligente. O poder da causa inteligente está na razão da grandeza do efeito.

—ALLAN KARDEC

RESUMO

Um dos grandes desafios em Sistema Multiagentes (SMA) é a criação de planos cooperativos para lidar com os diversos cenários que se apresentam num ambiente dinâmico, de tempo real, composto por times de robôs móveis. Neste cenário, cada robô é controlado por um agente do SMA, o qual precisa tomar decisões complexas em um curto espaço de tempo de forma coordenada com os demais robôs de seu time. Apesar das muitas soluções desenvolvidas com base em planejamento multiagente e aprendizagem por reforço, um espectador humano usualmente percebe oportunidades para melhores planos cooperativos em muitos cenários em que os robôs apresentam desempenho abaixo do esperado. A pesquisa apresentada nesta tese consiste em capturar o conhecimento do observador humano para demonstrar como times de robôs podem cooperar melhor na solução do problema que devem resolver. Como consequência, as diversas demonstrações humanas podem ser reunidas em um conjunto de dados para treinamento dos agentes que controlam os robôs. Para o desenvolvimento desta pesquisa, foi utilizado o ambiente *RoboCup 3D Soccer Simulation* (3DSSIM) e a coleta das demonstrações humanas foi realizada por meio de um conjunto de ferramentas desenvolvido a partir da adaptação de soluções existentes na comunidade RoboCup, utilizando uma estratégia de *crowdsourcing*. Além disso, foi utilizado o agrupamento fuzzy para reunir demonstrações que tenham o mesmo significado semântico, mesmo que com pequenas diferenças entre elas. Com os dados organizados, um mecanismo de aprendizagem por reforço foi utilizado para aprender uma política de classificação que permite aos agentes decidirem qual o grupo de jogadas é mais adequado a cada situação que se apresenta no ambiente. Os resultados evidenciam a capacidade de evolução do time de robôs, a partir da aprendizagem da política de seleção das jogadas sugeridas e do seu uso de forma adequada às habilidades de cada robô.

Palavras-chave: Sistemas Multiagentes; Aprendizagem por Reforço; Agrupamento Fuzzy; RoboCup; Futebol de Robôs.

ABSTRACT

One of the great challenges in *Multiagent Systems* (MAS) is the creation of cooperative plans to deal with the different scenarios that present themselves in a dynamic, real-time environment composed of teams of mobile robots. In this scenario, an agent of the MAS controls each robot, which needs to make complex decisions in a short time in a coordinated manner with the other robots on its team. Despite the many solutions developed based on multi-agent planning and reinforcement learning, a human observer usually sees opportunities for better cooperative plans in many scenarios where robots underperform. The research presented in this thesis consists of capturing the human spectator's knowledge to demonstrate how robot teams can better cooperate in solving the problem they must solve. The human watcher can indicate the situations in which a cooperative plan can better solve a given problem by watching the performance of a team of robots in action. Consequently, a dataset for training the agents that control the robots can gather the various human observations. For the development of this research, this work used the environment *RoboCup 3D Soccer Simulation* (3DSSIM) and the collection of human demonstrations was carried out through a set of tools developed from the adaptation of existing solutions in the RoboCup community using a strategy of crowdsourcing. In addition, fuzzy clustering was used to gather demonstrations with the same semantic meaning, even with small differences. With the data organized, this thesis used a reinforcement learning mechanism to learn a classification policy that allows agents to decide which group of plans is best suited to each situation that presents itself in the environment. The results show the ability of the robot team to evolve, from the learning of the suggested plays and its use in an appropriate way to the abilities of each robot.

Keywords: Multiagent Systems; Reinforcement Learning; Fuzzy Clustering; RoboCup; Robot Soccer.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

3DSSIM	*RoboCup 3D Soccer Simulation*	83
2DSSIM	*RoboCup 2D Soccer Simulation*	16
AFD	autômato finito determinístico	30
BahiaRT	*Bahia Robotics Team*	83
CBR	*Case-Based Reasoning*	15
DDPG	*Deep Deterministic Policy Gradient*	12
DQN	*Deep Q Network*	86
EM	*Expectation Maximization*	10
FCM	*Fuzzy C-Means*	73
FSF	*FCPortugal Setplays Framework*	73
HiTAB	*Hierarchical Training of Agent Behaviors*	13
IA	Inteligência Artificial	25
IVC	Índice de Validação de Clusters	86
LfD	*Learning from Demonstration*	40
MAS	*Multiagent Systems*	xiii
PPO	*Proximal Policy Optimization*	15
RBC	Raciocínio Baseado em Casos	86
SBSP	*Situation Based Strategic Positioning*	15
SF	Silhueta Fuzzy	73
SN	Silhueta Nítida (do inglês, *crisp silhouette)*	65
Sim-2D	Simulação 2D	25
Sim-3D	Simulação 3D	25
SMA	Sistema Multiagentes	83
SPlanner	*Strategy Planner*	84
SVM	*Support Vector Machine*	15

LISTA DE SÍMBOLOS

α Taxa de aprendizagem do agente executando aprendizagem Q

$\bar{\theta}$ Pesos utilizados em uma rede-Q para atualizar os valores alvo aproximados

$\bar{a}_{p,j}$ distância média do objeto j para todos os outros objetos *pertencentes* ao cluster p.

$\bar{B}_{i,k,z}$ número de comportamentos no vetor de comportamentos do z-ésimo passo do k-ésimo setplay do conjunto de dados ψ_i gerado no nível 1 do agrupamento fuzzy.

$\bar{b}_{p,j}$ o $d_{q,j}$ mínimo calculado sobre $q = 1, \ldots, c$, $q \neq p$

$\bar{t}$ tempo limite transcorrido desde o início de um episódio no treinamento da rede-Q com a política de seleção de setplays.

$\beta_{i,k,z,j}$ string que representa o j-ésimo comportamento no vetor de comportamentos do z-ésimo passo do k-ésimo setplay do conjunto de dados ψ_i gerado no agrupamento fuzzy de nível 1

$\ddot{a}^i$ Número de agentes participantes do passo ou estado S_i de um setplay, onde $i = 0, \ldots, \ddot{m} - 1$

$\ddot{m}$ Total de papéis, passos ou estados de um setplay

$\ddot{m}_k$ quantidade de passos do k-ésimo setplays em um conjunto de dados.

Δ_j Acumulador da quantidade de comportamentos diferentes entre dois vetores de comportamentos.

γ Fator de desconto que define a preferência de um agente em relação a recompensas imediatas e futuras

$\hat{Q}$ Função ação-valor alvo usada no algoritmo DQN

$\hat{t}$ Instante discreto no tempo durante a vida de um agente baseado em aprendizagem Q

κ^i Total de transições em um passo ou estado S_i de um setplay

λ Coeficiente de ponderação utilizado no cálculo da silhueta fuzzy.

π Política usada por um agente para selecionar a próxima ação $a_{\hat{t}}$ a parte do estado atualmente observado $s_{\hat{t}}$

Ψ Probabilidade para escolha de uma ação aleatória no algoritmo DQN

ψ_i Conjunto de instâncias do grupo i pertencente ao dataset agrupado 1; $i = 1, \ldots, C^{(1)}$; $\psi_i \in GD^{(1)}$

σ_j silhueta do objeto j no conjunto de dados.

τ_i Conjunto de condições de transição do passo ou estado S_i

θ Pesos de uma rede-Q

ε Limite para a melhoria entre as partições *fuzzy* da iteração atual e anterior no algoritmo *Fuzzy C-Means* (FCM)

ζ Total de condições de término com falha de um setplay

A Conjunto de ações que um agente é capaz de executar. Também recebe o nome de espaço de ação

a uma ação $a \in A$ qualquer pertencente ao espaço de ações do agente

$a_{\hat{t}}$ Ação selecionada por um agente para ser executada no instante $\hat{t}$

B_i Conjunto de comportamentos dos agentes participantes do passo ou estado S_i de um setplay, onde $i = 0, \ldots, \ddot{m} - 1$

b_{ij} Comportamento associado ao agente j no passo ou estado S_i de um setplay, onde $j = 1, \ldots, \ddot{a}^i$ e $i = 0, \ldots, \ddot{m} - 1$

C Quantidade de passos de tempo para reinicializar a função ação-valor alvo $\hat{Q}$ com os pesos da função ação-valor Q o algoritmo DQN

c Número de grupos ou *clusters* em um agrupamento Fuzzy

$C^{(1)}$ Quantidade de grupos ou clusters na partição fuzzy resultante do algoritmo FCM no primeiro nível.

$C^{(2)}$ Quantidade de grupos ou clusters na partição fuzzy resultante do algoritmo FCM no segundo nível.

C_I Condição inicial para ativar um setplay

C_s Condição de término com sucesso de um setplay

C_{fi} Condições de término com falha de um setplay, onde $i = 1, \ldots, \zeta$

d Função que mede a dissimilaridade entre os elementos no processo de agrupamento *fuzzy*

$d_{q,j}$ a distância média do objeto j para todos os objetos *pertencentes* a outros clusters q

$GD^{(1)}$ Conjunto de dados agrupados no primeiro nível do FCM

J_{FCM} Função objetivo do algoritmo FCM

m Expoente de ponderação *fuzzy* ou *fuzzificador*

N Limite máximo para a quantidade de experiências na memória de repetição do algoritmo DQN

n Quantidade de objetos em X

O Conjunto de estados distintos que descrevem o ambiente de um agente utilizando aprendizagem Q. Também recebe o nome de espaço de observação

p Dimensão dos objetos em X

P_B Conjunto de jogadores do *Bahia Robotics Team* (BahiaRT) que participam de um passo de um setplay.

p_B Um jogador do time BahiaRT; $p_B \in P_B$.

P_O Conjunto de jogadores do time oponente que participam de um passo de um setplay.

p_O Um jogador do time adversário; $p_O \in P_O$.

Q Função ação-valor usada nos algoritmos Q-learning e DQN

r Função de recompensa de um agente baseado em aprendizagem Q

R_{sp} Conjunto de papéis de um setplay

S Conjunto de passos ou estados de um setplay

s Um estado $s \in O$ qualquer pertencente ao espaço de observação do agente

S_i um passo ou estado de um setplay, onde $i = 0, \ldots, \ddot{m} - 1$ e $S_i \in S$

$s_{\hat{t}}$ Estado observado atualmente por um agente no instante $\hat{t}$

SP_i Um setplay i qualquer.

spr_i Um papel de um setplay, onde $i = 1, \ldots, \ddot{m}$ e $spr_i \in R_{sp}$

T_{ij} Transição j no passo ou estado S_i de um setplay, onde $j = 1, \ldots, \kappa^i$

t_{max} Número máximo de iterações do algoritmo FCM

u_{ik} Grau de pertinência do objeto x_k no i-ésimo grupo, onde $1 \leq k \leq n$; $1 \leq i \leq c$; $u_{ik} \in [0, 1]$

V Conjunto de centróides dos grupos em uma partição *fuzzy*

$V^{\pi}(s_{\hat{t}})$ Recompensa cumulativa descontada obtida por um agente aplicando repetidamente a política π a partir do estado $s_{\hat{t}}$

v_i Centróide do i-ésimo grupo em uma partição *fuzzy*, onde $1 \leq i \leq c; v_i \in V$

X Conjunto de objetos(*dataset*) em um espaço p-dimensional ($X \subset \mathbb{R}^p$)

x_k Objetos em X, onde $k = 1, \ldots, n$

$y_{\hat{t}}$ Valores alvo aproximados na iteração $\hat{t}$ do algoritmo DQN

$y_{i,k,z}$ z-ésimo passo no k-ésimo setplay no conjunto de dados ψ_i gerado no nível 1 do agrupamento fuzzy.

$Y_{i,k}$ conjunto de dados formado pelas propriedades do k-ésimo setplay no cluster ψ_i gerado pelo primeiro nível de agrupamento fuzzy.

t Número da iteração atual no algoritmo FCM

U Partição *fuzzy* ou matriz de graus de pertinência

Capítulo 1

Neste capítulo, a contextualização do tema, apresentação do problema de pesquisa e objetivos deste trabalho são apresentados.

INTRODUÇÃO

A Inteligência Artificial (IA) é um tema que gradativamente vai saindo dos círculos fechados das Universidades e laboratórios de pesquisa e ganha espaço e interesse do grande público, com repercussão relevante nas mídias de massa. Um dos conceitos base da IA é a autonomia que um artefato inteligente precisa possuir. Este artefato é denominado agente inteligente (RUSSELL; NORVIG, 2021). Popularmente, tem-se chamado estes mesmos artefatos de robôs, independente de serem de fato robôs físicos, capazes de atuar em ambientes reais, ou robôs de software (também conhecidos como *softbots*), que atuam em ambientes virtuais (como a Internet ou um Simulador). Neste trabalho, os termos "robô"e "agente"serão utilizados de forma intercambiável para descrever um artefato inteligente de qualquer natureza.

Não há dúvidas que vive-se uma rápida transição para uma sociedade em que humanos e robôs conviverão no dia a dia de forma natural e corriqueira. Notícias de desenvolvimento de veículos autônomos (carros sem motorista, *drones* para entregas de mercadorias, etc), robôs capazes de realizar movimentos em ambientes hostis e outros tantos capazes de interagir com humanos de forma cordial e gentil, robôs que cuidam dos investimentos das pessoas na bolsa de valores, ou ainda fazem compras na internet ou num supermercado, recheiam as notícias da grande mídia. Este cenário fomenta uma intensa atividade científica na área de IA, em especial em aprendizagem de máquina. A aprendizagem de máquina supre uma importante demanda dos robôs por aprender comportamentos e conhecimentos intuitivos ou mesmo instintivos do ser humano. Por exemplo, apesar do movimento "andar" ser natural ao ser humano, o conhecimento algorítmico de como o organismo o executa não é conhecido. A aprendizagem de máquina permite que robôs consigam andar de forma muito semelhante a um ser humano, aprendendo a generalizar os movimentos a partir de exemplos. Sendo assim, a próxima seção descreverá a contextualização desta área de pesquisa, bem como a motivação para realização deste projeto.

1.1 CONTEXTUALIZAÇÃO E MOTIVAÇÃO

Diante dos mais diversos avanços em IA e robótica, grupos de cientistas de diversas partes do mundo têm considerado os robôs como um dos grandes desafios científicos para o século atual. Para atender às atuais demandas da sociedade, robôs devem ser capazes de interagir com humanos e também uns com os outros de forma a agir cooperativamente em uma sociedade de robôs. Estas características são apresentadas no conceito de Sistema Multiagentes (SMA). (WEISS, 1999).

SMAs são sistemas compostos por múltiplos elementos computacionais interativos conhecidos por agentes (WOOLDRIDGE, 2009). Um agente é um elemento capaz de perceber seu ambiente através de sensores e agir sobre este mesmo ambiente através de atuadores, conforme ilustrado na Figura 1.1. Um agente é considerado inteligente ou autônomo se possuir uma característica fundamental: autonomia. Um agente é autônomo se é capaz de decidir suas ações por conta própria, sem intervenção humana (RUSSELL; NORVIG, 2021). Outra característica fundamental para que um agente possa fazer parte de um SMA é a capacidade de interagir com outros agentes. A interação não se restringe à troca de informações, deve incluir algum tipo de atividade social como cooperação, coordenação, negociação, etc (WOOLDRIDGE, 2009).

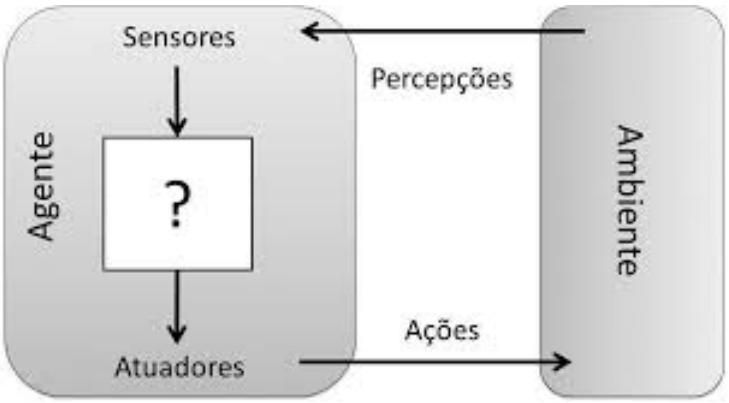


Figura 1.1: Esquema ilustrativo da definição de agente.

Os problemas que demandam grupos ou times de robôs para serem resolvidos (ex: entrega de mercadorias, produção industrial, construção civil, robôs de serviços, etc) podem ser modelados como um SMA. SMA podem ser organizados em diversas arquiteturas. Cada robô pode ser modelado por um agente do SMA ou ter vários agentes controlando partes diferentes de um mesmo robô. Em qualquer arquitetura, o problema a ser solucionado pelo SMA é modelado como o ambiente dos agentes que compõem o SMA.

O ambiente pode ter algumas propriedades que permitem definir o nível de complexidade para sua solução. Ambientes podem ser totalmente observáveis ou parcialmente observáveis. Em ambientes totalmente observáveis, todas as propriedades relevantes do ambiente para solução do problema estão sempre disponíveis para os agentes através de seus sensores. Nos ambientes parcialmente observáveis, parte das propriedades não estão sempre disponíveis, podendo estar disponíveis em alguns instantes e inacessíveis em

outros instantes. O ambiente pode ser determinístico ou estocástico. No ambiente determinístico o agente pode sempre prever qual será o efeito exato de suas ações antes mesmo de executá-las. O ambiente estocástico é não-determinístico, ou seja, não é possível sempre prever qual será o efeito preciso de cada ação executada. Em ambientes estocásticos, os agentes só poderão ter certeza de qual o efeito das suas ações após executá-las, se estes efeitos estiverem disponíveis como percepções para seus sensores. O ambiente pode ainda ser estático ou dinâmico. Num ambiente estático, os valores das propriedades obtidas por percepção pelos agentes não se modifica enquanto os agentes deliberam qual ação ou sequencia de ações irão executar. Nos ambientes dinâmicos, os valores destas propriedades percebidas podem modificar-se a todo instante, mesmo enquanto o agente está deliberando as suas próximas ações. Fica claro que os ambientes parcialmente observáveis, estocásticos e dinâmicos apresentam uma complexidade maior que outros ambientes totalmente observáveis, determinísticos e estáticos (RUSSELL; NORVIG, 2021).

No contexto dos SMA, além das possíveis propriedades descritas, um ambiente pode ser considerado de tempo real. Um ambiente de tempo real é aquele em que determinadas tarefas ou ações possuem prazos rígidos para serem executados ou não terão mais utilidade ou efeito desejável. Para efeito ilustrativo, considere um SMA formado por robôs que precisam cooperar para construir uma casa. Dois robôs trabalham de forma coordenada para levantar as paredes da casa. Um robô empilha blocos, enquanto outro robô irá colocar concreto entre os blocos. Se o robô que coloca o concreto atrasar sua tarefa, o primeiro robô irá empilhar dois blocos sem que tenha concreto entre eles. Desta forma, será inútil que o segundo robô coloque o concreto depois que os dois blocos estiverem empilhados sem a massa de concreto que irá fazer a liga entre os blocos. Ou seja, há um prazo claro para que uma ou mais tarefas sejam concluídas, caracterizando um ambiente de tempo real.

Toda a complexidade de problemas reais em ambiente distribuído, de tempo real, dinâmico, parcialmente observável e estocástico pode ser reduzido ao desafio padrão, eleito pelos cientistas, do futebol de robôs(KITANO et al., 1998). Para vencer este desafio, robôs devem ter a capacidade de tomar decisões complexas em um espaço de tempo muito reduzido, cooperando com robôs aliados e competindo contra os oponentes robôs ou humanos. Desde 1997, a RoboCup Federation[1] promove o desenvolvimento científico em inteligência artificial e robótica através de competições científicas entre robôs. O evento anual *Robot World Cup*(RoboCup) reúne um simpósio científico e uma série de desafios práticos para times de robôs projetados nos laboratórios científicos ao redor do mundo provarem na prática os resultados das suas pesquisas.

SMA projetados para solucionar o problema do futebol de robôs precisam ter agentes capazes de lidar com comportamentos chamados de baixo nível (executar movimentos como andar ou rodar, chutar, levantar-se, etc). Também precisam tomar decisões com nível mais abstrato. Por exemplo, não basta saber como controlar motores que façam o robô rodar ou andar, é necessário decidir qual caminho seguir. Se o robô estiver com a posse de bola, ainda deve garantir que não perca esta bola enquanto se desloca. Portanto, decisões mais abstratas envolvendo planejamento de caminhos e desvio de obstáculos

[1] http://www.robocup.org

(outros robôs do mesmo time ou oponentes, traves, etc). Um outro nível de abstração mais alto é o que demanda coordenação e cooperação entre os agentes do SMA. Um time de robôs sem coordenação pode resultar num cenário em que todos os robôs do time tentem ir até a bola ao mesmo tempo, colidindo uns com os outros e, consequentemente, perdendo a posse de bola para o adversário que terá espaços vazios para marcar facilmente um gol. Os agentes precisam, portanto, coordenar-se para garantir que apenas um agente tente manter a posse de bola. Por outro lado, precisam evitar um estado em que nenhum agente tente manter ou obter a posse de bola, pois este comportamento coletivo entregaria a bola facilmente ao time oponente, o que também é indesejado para os objetivos do SMA. Comportamentos como comunicação e capacidade de percepção como audição e visão são essenciais para que todas estas camadas de abstração possam ser tratadas pelos agentes em um SMA de robôs jogadores de futebol.

Costa e Bittencourt (2000) apresentaram o Agente Autônomo Concorrente: um modelo para desenvolver agentes capazes de lidar com todas estas camadas de abstração necessárias para um agente jogador de futebol. O modelo foi utilizado de forma bem sucedida em alguns desafios de futebol de robôs (COSTA; BITTENCOURT, 1999)(Júnior, Orivaldo Vieira Santana; da Costa, Augusto Loureiro, 2006). O uso do agente autônomo concorrente representa um grande avanço para viabilizar o desenvolvimento de SMA robóticos coordenados e cooperativos. Entretanto, o agente autônomo concorrente não é suficiente para que os agentes de fato cooperem para a criação de jogadas coletivas complexas. Ao analisar jogos reais de futebol, percebe-se que há uma diversidade de jogadas coletivas complexas que são criadas e adaptadas por treinadores e jogadores de forma contínua. A motivação para a criação de jogadas é encontrar posicionamento e movimentos dos jogadores que sejam capazes de confundir o time oponente maximizando as chances de marcar gols e minimizando as possibilidades de conceder gols. No futebol de robôs, esta mesma necessidade se faz presente. Jogadas coletivas ou *setplays* são formalizados através de planos cooperativos num SMA.

Pesquisas recentes indicam grande interesse na criação de planos cooperativos para times de robôs que atuam em conjunto na resolução de um certo problema (D'AMBROSIO; STANLEY, 2013; ZHANG; TAMBE, 2015; ZHANG et al., 2017; YU et al., 2015b; LIEMHETCHARAT; VELOSO, 2017). Outros trabalhos utilizam, como ambiente de validação dos resultados de suas pesquisas com planos cooperativos, o desafio de futebol de robôs, como proposto pela RoboCup Federation (MOTA; REIS; LAU, 2011; CRAVO et al., 2014; FABRO; REIS; LAU, 2014; ALMEIDA et al., 2013; FREELAN et al., 2014; BIANCHI et al., 2018; SHI et al., 2018). Os resultados demonstram uma clara necessidade de utilizar métodos de aprendizagem de máquina para permitir que os robôs em um SMA aprendam planos intuitivos, baseados no conhecimento não estruturado do observador humano. A principal questão em aberto está relacionada à forma como robôs podem aprender e adaptar planos cooperativos às demandas de ambientes complexos, como o futebol de robôs. Atualmente, cientistas e desenvolvedores criam manualmente os *setplays* tentando prever todas as condições possíveis para seu uso e os comportamentos de cada agente durante sua execução. Esta abordagem melhora o comportamento do time de robôs jogadores de futebol se comparado a um modelo meramente reativo ou sem coordenação. Entretanto, é usual perceber os criadores dos *setplays* criticando o

comportamento dos seus times em diversas situações. Atribui-se este fato à dificuldade de generalizar o conhecimento intuitivo sobre o domínio do problema (futebol). Portanto, há uma motivação clara para aperfeiçoar as abordagens já utilizadas de aprendizagem de máquina para tentar capturar este conhecimento do espectador humano para permitir a aprendizagem de *setplays* completos.

Durante o desenvolvimento de *setplays*, é comum utilizar informações imprecisas ou mesmo incompletas. Por exemplo, para um agente robótico envolvido numa jogada coletiva de ataque decidir se passa a bola para outro robô de seu time ou não, precisa levar em consideração a posição dos robôs do seu time, dos robôs do time oponente, da bola, dos gols de seu time e do time oponente, entre outras informações. Entretanto, utilizando um sistema de visão parcial (com câmera embarcada no corpo do robô que fornece uma visão apenas parcial do campo), nem sempre o robô terá todas estas informações disponíveis. O robô pode ainda ter algumas informações desatualizadas, por exemplo, localização de algum dos objetos visualizadas há algum tempo no passado que já não corresponde mais à localização atual deste objeto. No mesmo exemplo, o robô ainda irá decidir se vai fazer o passe ou continuar conduzindo a bola a depender da posição do oponente mais próximo. Se este oponente estiver muito próximo, o passe será realizado, caso contrário continuará conduzindo a bola. O conceito de **muito próximo** não é preciso. Em alguma situações, distâncias inferiores a 1 m serão consideradas **muito próximas**. Em outras situações, distâncias inferiores a 3 m já podem ser tratadas como **muito próximas**. Isto dá um caráter adjetivo a algumas propriedades numéricas que é mais importante que o valor exato destas propriedades. Nestes exemplos, nem sempre o uso de um valor arbitrário ou definido empiricamente irá ser suficiente para generalizar a situação que se faz necessária. Outros conceitos como posse de bola, passes certos, e etc, também não possuem definições facilmente mensuráveis de forma exata com base nas informações disponíveis no ambiente de futebol de robôs. Desta forma, um ponto de investigação que se faz necessário ao aprender novos *setplays*, é como representar esta informação incerta ou imprecisa.

A próxima subseção apresenta os objetivos deste trabalho e abordagem desenvolvida para solucionar o problema apresentado.

1.2 OBJETIVOS

Este trabalho tem como objetivo apresentar evidências experimentais de que é possível capturar o conhecimento intuitivo ou não estruturado de humanos ao assistir a um jogo de robôs jogando futebol para compor um conjunto de dados para treinamento do time de robôs. O treinamento permite aprender uma política de seleção de jogadas coletivas capaz de influenciar positivamente o desempenho global do time. Jogadas coletivas coordenadas, chamadas de setplays, compõem o conjunto de dados. Utilizou-se uma solução de aprendizado de reforço profundo para que a equipe aprenda uma política de seleção de setplays a partir de um grande conjunto de demonstrações realizadas por humanos.

Para atingir este objetivo, esta tese fornece ao espectador uma interface em que possa assistir partidas de futebol de robôs, analisando-as criticamente e gerando recomendações de jogadas coletivas que poderiam ter sido utilizadas pelos robôs em situações diversas da partida para obterem um melhor desempenho. O usuário fará, portanto, o papel

semelhante a um treinador ou analista no ambiente real do futebol.

Mapeou-se as informações que servem de base para representar o conhecimento do espectador. Este mapeamento respeita o eventual caráter de imprecisão ou indisponibilidade total ou parcial de algumas informações num dado instante. Um objetivo adicional desta tese é definir um modelo de particionamento *fuzzy* para representar as informações imprecisas ou incompletas que façam parte do conjunto de dados gerado pelo demonstrador. Desta forma, o demonstrador não precisa definir valores exatos criados artificialmente ou arbitrariamente para representar os conceitos intuitivos que pretende demonstrar.

A solução gerada contempla a possibilidade de obter conhecimento de diversas fontes distintas. Portanto, diversos espectadores diferentes poderão fornecer suas recomendações a partir dos mesmos jogos analisados ou de jogos diferentes. Há ainda a possibilidade de um demonstrador gerar duas ou mais recomendações que tenham valor semântico equivalente numa mesma partida analisada ou em partidas distintas. Por equivalência semântica, entende-se duas recomendações cujos valores de propriedades do ambiente sejam diferentes mas que representem uma mesma jogada coletiva ou uma mesma situação de ativação de uma jogada coletiva. Considerando como exemplo as situações ilustradas na Figura 1.2, pode-se notar que os dois cenários ilustrados representam semanticamente o mesmo *setplay*. Na situação da Figura 1.2a, os jogadores 2 e 3 irão avançar conforme indicado nas setas azuis no primeiro estágio do *setplay*. No estágio seguinte, o jogador 1 fará um passe em direção à posição esperada do jogador 2, como indicado pela seta vermelha. A sequência da jogada deve terminar com o jogador 2 passando a bola para o jogador 3 completando a triangulação, jogada comum no futebol. Na situação da figura 1.2b, os jogadores 2, 3 e o oponente (círculo vermelho) estão em posições distintas das que estavam na situação (a). As distâncias relativas entre os objetivos também são diferentes daquelas refletidas na situação (a), mas a jogada coletiva é a mesma: triangulação. Desta vez, após os jogadores 2 e 3 avançarem, o jogador 1 fará um passe para o jogador 3 que posteriormente completará a triangulação passando a bola para o jogador 2.

As instâncias do conjunto de dados (*dataset*) gerado pelas recomendações dos espectadores nos dois cenários descritos na Figura 1.2 terão valores de suas propriedades diferentes (localização dos objetos, distâncias relativas, etc). Mas as duas instâncias representam uma mesma jogada coletiva e são consideradas, portanto, semanticamente equivalentes. Esta tese investigou métodos para identificar esta equivalência semântica, com o objetivo de otimizar o conjunto de dados de treinamento antes de treinar o SMA para aprender novos *setplays*. A solução apresentada é capaz de lidar com *datasets* potencialmente grandes gerados por vários espectadores, além de evitar redundância desnecessária em exemplos de treinamento.

A solução final apresentada nesta tese utiliza o dataset formado por demonstrações coletadas massivamente através da solução de coleta BahiaRT Setplays Collecting Toolkit, desenvolvida neste trabalho, organizada pela solução de particionamento difuso também produzida nesta tese. O dataset foi utilizado como base do modelo de aprendizagem por reforço profunda para prover ao SMA uma política de seleção de *setplays* adequada às necessidades e características dos agentes que compõem o SMA.

(a) Primeiro exemplo de *Setplay* (b) Segundo exemplo de *Setplay*

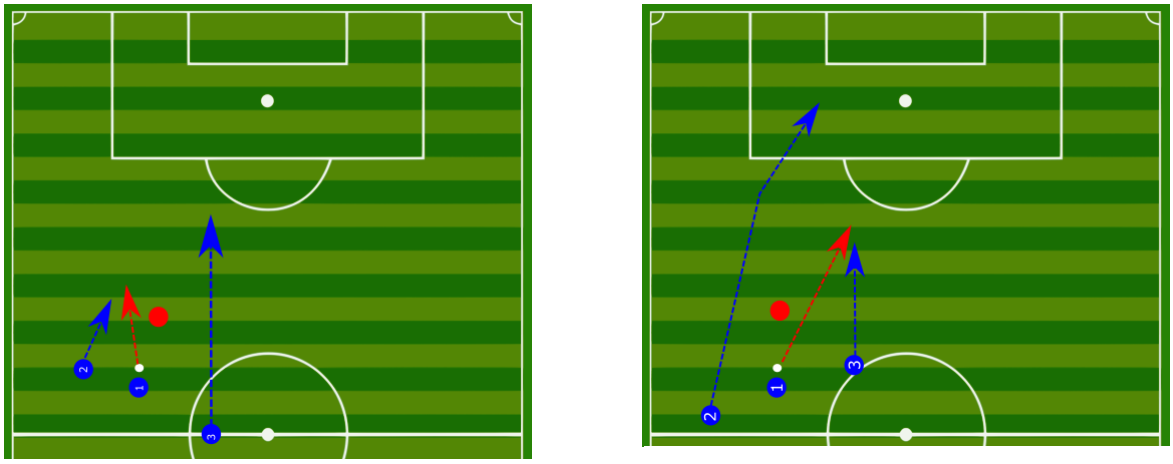


Figura 1.2: Exemplos de *Setplay* para ilustrar o conceito de equivalência semântica. O primeiro *setplay* (a) representa uma triangulação para ultrapassar o oponente e aproximar-se da área. O segundo *setplay* (b) é o mesmo tipo de jogada, com o mesmo objetivo, mas ações diferentes.

Os resultados desta tese evidenciaram a melhoria no desempenho do SMA, comprovando a hipótese apresentada. A próxima seção descreve a organização do restante deste trabalho.

1.3 ORGANIZAÇÃO DO TRABALHO

O Capítulo 2 apresenta o levantamento do estado da arte realizado durante esta pesquisa. Os principais trabalhos relacionados são analisados evidenciando sua contribuições e identificando as lacunas presentes na literatura. O capítulo 3 apresenta os conceitos básicos necessários ao desenvolvimento deste trabalho. A descrição do desafio de futebol de robôs com todo o ambiente RoboCup é apresentado, juntamente com o conceito de *setplays*. Os conceitos e algoritmos de agrupamento fuzzy e aprendizagem por reforço também estão explicados neste capítulo. A solução desenvolvida nesta tese, com descrição detalhada de todos os produtos gerados e da metodologia utilizada, está no capítulo 4. Os resultados experimentais que comprovam os objetivos do trabalho estão descritos no capítulo 5. A discussão dos resultados e das contribuições da tese, além da análise das possibilidades futuras de pesquisa que este trabalho permite estão no capítulo 6.

Capítulo

2

Neste capítulo, apresenta-se a revisão da literatura com os principais trabalhos que constituem o estado da arte no tema investigado.

TRABALHOS RELACIONADOS

Para fundamentar o presente projeto, foi realizado o levantamento do estado da arte sobre aprendizagem de máquina em planejamento de Sistemas Multiagentes (SMA) robóticos, coletando informações acerca de pesquisas envolvendo a aprendizagem automática de planos multiagentes com ênfase no domínio de esportes coletivos (*setplays*). Também foi realizado um levantamento sobre trabalhos que relatam experiências em domínios com informações incertas, imprecisas ou incompletas. A investigação incluiu, de forma complementar, o levantamento de trabalhos que tratam do agrupamento de instâncias que apresentam equivalência semântica em um conjunto de dados de treinamento para um SMA. Entende-se que duas instâncias em um conjunto de dados são equivalentes semanticamente se representam um mesmo plano ou situação de ativação de plano, mesmo com valores diferentes em suas propriedades.

Para orientar este estudo definiu-se algumas questões de pesquisa para serem respondidas:

1. Quais abordagens de aprendizagem de máquina são utilizadas para aprendizagem de planos de ação em sistemas multiagentes?

2. Quais abordagens de aprendizagem por demonstração são utilizadas em sistemas multiagentes robóticos?

3. Quais abordagens de agrupamento (aprendizagem de máquina não supervisionada) são usadas para tratamento de instâncias semanticamente similares em *datasets* grandes?

4. Quais abordagens para tratamento de incerteza ou imprecisão são utilizadas em sistemas multiagentes robóticos?

Neste capítulo, os resultados da revisão de literatura são apresentados divididos em seções que tratam de i) planejamento em SMA (seção 2.1); ii) *setplays* em futebol robótico (seção 2.2) e iii) agrupamento de instâncias semanticamente equivalentes em datasets (seção 2.3). Conclui-se o capítulo discutindo as informações levantadas no estado da arte e reforçando as contribuições deste trabalho para o avanço nesta área(seção 2.4).

2.1 PLANEJAMENTO EM SMA

O planejamento em SMA tem na coordenação entre os agentes uma de suas principais características (WOOLDRIDGE, 2009). Para compreender, como o planejamento de ações coordenadas é realizado em SMA, realizou-se um levantamento dos principais trabalhos recentes na área, buscando identificar as lacunas no que diz respeito à adição do conhecimento não formalizado do especialista humano.

D'Ambrosio e Stanley (2013) apresentam uma solução escalável para times muito grandes (aprox. 1024 agentes) onde o aprendizagem multiagentes considera padrões geométricos do time ao invés de agentes individuais. Desta forma a partir do posicionamento relativo dos agentes entre si, são identificados padrões de comportamento associados a papéis que são definidos conforme o posicionamento do agente. Este padrão é generalizado através de um modelo de redes neurais denominado HyperNEAT. Entretanto, os autores afirmam que sua principal contribuição é no modelo conceitual. Não foi realizada qualquer validação em problemas reais com SMA ou desafios como futebol robótico. O foco do trabalho é a escalabilidade e evitar duplicação de aprendizagem de um mesmo conhecimento por vários agentes distintos. Não há preocupação explícita com as questões relacionadas a coordenação dos comportamentos em cada passo e condições de início, término e transições.

Zhang e Tambe (2015) descrevem o uso de redes bayesianas para aprender o comportamento de criminosos num framework de planejamento de unidades de patrulhamento criminal em uma área urbana. O modelo leva em conta as mudanças no mundo, inclusive da localização dos agentes em tempo real, para modelar a decisão de criminosos oportunistas em executar um crime num dado instante e local. Posteriormente, Zhang, Sinha e Tambe (2015) apresentam resultados positivos de um framework para modelar o comportamento de adversários (especificamente criminosos oportunistas numa área urbana) utilizando redes bayesianas junto com o algoritmo *Expectation Maximization* (EM). Estes trabalhos não utilizam a aprendizagem de máquina para aprender planos em SMA e sim comportamento dos oponentes para dar suporte ao planejamento. Panella e Gmytrasiewicz (2017) também utilizam a aprendizagem do comportamento de outros agentes como suporte ao planejamento em um SMA. O aprendizagem é intercalado com o planejamento para alimentar os planos gerados. A solução é modelada com cadeias de Markov utilizando métodos de Monte Carlo para o aprendizagem. Não há a aprendizagem de planos completos propriamente ditos e sim da influencia dos possíveis planos/comportamentos de outros agentes no plano individual de cada agente do SMA. Não há a influência de conhecimento de um especialista humano no processo de aprendizagem.

Yu et al. (2015b) apresentam um novo modelo para representar a concorrência no planejamento em um SMA utilizando-o com duas abordagens de aprendizagem: Monte

Carlo e *Learning from Demonstration* (LfD). Entretanto, a validação deste trabalho foi feita num domínio de robôs estáticos manipuladores que precisam cooperar na montagem de um objeto. Os requisitos de cooperação em ambientes de robôs móveis em tempo real, estocástico e parcialmente observável apresentam uma ordem de complexidade superior ao domínio que os autores usaram para validar seu trabalho. Na proposta de uso do conhecimento dos especialistas humanos através do LfD não fica claro de que forma instâncias semanticamente equivalentes geradas por diversos especialistas seriam agrupadas para otimizar o tamanho do *dataset*. É possível que o domínio usado para validação tenha uma baixa dimensionalidade e os tamanhos dos *datasets* gerados não sejam um problema. Em outro trabalho (YU et al., 2015a), um framework baseado em aprendizagem por reforço multiagentes utiliza emoções como fator de motivação para resolver o dilema social evitando o equilíbrio de Nash. A situação de equilíbrio de Nash é modelada, com base na teoria de jogos, como um dilema do prisioneiro (RAPOPORT; CHAMMAH; ORWANT, 1965). Aplicando este conceito a aprendizagem multiagentes, dois agentes tendem a ter maior recompensa individual se escolherem ações individualistas, que evitam a cooperação. Este estado é o equilíbrio de Nash que não representa a solução cooperativa ótima. Portanto, Yu et al. (2015a) utilizam modelagem de emoções nos agentes para permitir que a aprendizagem se afaste do equilíbrio de Nash e possa convergir para a solução de cooperação ótima. Apesar de descrever o uso importante de fatores de emoção para tomada de decisões cooperativas, não há uma clara aplicação a planejamento em SMA aplicado a um problema real com características semelhantes ao domínio do futebol robótico.

Zhou, Purvis e Muhammad (2015) descrevem uma combinação do algoritmo *Q-Learning* com Redes Bayesianas distribuídas para modelar o planejamento de cadeias de suprimento (*supply chains*) num mercado global de compra e venda de produtos. Apesar dos bons resultados no domínio de problema explorado, esta abordagem não realiza aprendizagem de planos cooperativos completos. A própria natureza do problema difere bastante do ambiente de um SMA formado por múltiplos robôs móveis. Também não é utilizado o conhecimento de um especialista humano como nas propostas baseadas em LfD. A interdependência entre o comportamento dos agentes também é tema do trabalho de Micalizio e Torta (2016). O foco dos autores concentra-se no diagnóstico de planos em SMA para detectar falhas (e suas causas) em ações associadas ao atraso em ações interdependentes. Um formalismo utilizando inferência bayesiana foi desenvolvido e validado. Mas apenas a inferência sobre o conhecimento existente é utilizada, não há aprendizagem de novos conhecimentos ou planos. A contribuição deste trabalho pode ser utilizada como parte da proposta apresentada neste projeto envolvendo o aprendizagem de planos completos que incluem as condições de cancelamento por falha (*abort*).

Zhang et al. (2017) definem um modelo para planejamento simbólico considerando um algoritmo iterativo para otimização dos planos. Os planos otimizados evitam conflitos e melhoram a sinergia das ações. Adicionalmente, os autores definem um modelo probabilístico com uma distribuição de Poisson para representar a incerteza temporal da navegação em função do aparecimento de obstáculos. O problema geral de navegação de robôs foi utilizado como ambiente para validação da proposta. Além dos robôs neste SMA possuírem um único comportamento possível (navegação), o ambiente é conhecido

e possui baixo nível de não-determinismo. A única incerteza presente refere-se às ações dos outros agentes e eventuais obstáculos móveis. Não há múltiplos comportamentos distintos possíveis a cada instante nem a necessidade de cooperação para que cada robô alcance seu objetivo. Liemhetcharat e Veloso (2017) apresentam uma solução para treinar em tempo real a coordenação entre agentes usando um conceito de Grafos de Sinergia. A proposta foi validada em um SMA genérico aplicado a um problema teórico bem conhecido (*multi-armed bandit problem*). Não há validação em problemas reais, como o futebol de robôs. Não está claro em que nível o aprendizagem acontece, se restrito à cooperação ou se envolve o aprendizagem de planos multiagentes completos.

O levantamento encontrou um trabalho que utiliza os algoritmos de aprendizagem por reforço com redes profundas para aprendizagem de habilidades e estratégias para jogos de precisão individuais, como a bocha (ANTãO et al., 2020). Os autores construíram um simulador e usaram o algoritmo *Deep Deterministic Policy Gradient* (DDPG) com um braço robótico com sete graus de liberdade. Os resultados foram animadores e demonstraram mais uma vez a importância do uso da aprendizagem por reforço profunda para aprendizagem de habilidades em sistemas robóticos.

Utilizando uma abordagem de comportamentos coletivos emergentes, Miyashita e Sugawara (2020) encontraram evidências de que comportamentos coletivos podem emergir em um SMA em que cada agente use sua própria *Deep Q Network* (DQN). Os experimentos foram conduzidos utilizando o problema coleta e distribuição de materiais. Neste problemas, um time de robôs deve coletar materiais em áreas de armazenamentos e distribui-los em espaços vazios num ambiente. É uma abstração de um ambiente onde robôs precisem agir de maneira coordenada carregando materiais pesados num canteiro de obras ou instalação industrial, por exemplo.

Até o momento, nenhum dos trabalhos apresentados demonstrou validação de seus resultados através de planejamento em futebol robótico. Também não foram encontradas soluções que buscam capturar o conhecimento intuitivo do especialista no domínio de problema para alimentar o dataset de treinamento do SMA. Na próxima seção, trabalhos voltados para *setplays* em futebol de robôs são apresentados.

2.2 SETPLAYS EM FUTEBOL ROBÓTICO

Um dos grandes marcos para o desenvolvimento de *setplays* em futebol robótico foi a proposta de um framework para facilitar a criação destas jogadas coletivas (MOTA; LAU; REIS, 2010). O framework torna mais fácil e rápida a criação de *setplays* utilizando uma linguagem de mais alto nível que as linguagens de programação comumente utilizadas. A linguagem proposta é uma boa base para criação de *setplays*, seja diretamente por um especialista humano ou por um mecanismo de aprendizagem de máquina. Reis et al. (2010) complementam o trabalho anterior acrescentando uma interface gráfica para definir formações estratégicas e *setplays*. A ferramenta apresentada gera *setplays* representados na mesma linguagem definida por Mota, Lau e Reis (2010). Nenhum destes trabalhos utiliza aprendizagem de máquina, restando ao desenvolvedor todo o trabalho de criação do *setplay*. A criação individual de *setplays* lida com o problema de alta dimensionalidade de situações possíveis do ambiente de futebol de robôs. Considerando que é impraticável

prever todas as situações possíveis no domínio de futebol de robôs, o conhecimento de especialistas em futebol seria muito útil para detectar, a partir de jogos reais, falhas cometidas pelos times de robôs. Os especialistas podem sugerir comportamentos mais desejáveis para estes cenários de falha. As soluções apresentadas (MOTA; LAU; REIS, 2010; REIS et al., 2010; ALMEIDA et al., 2013) não dão suporte a este tipo de intervenção do especialista humano.

Almeida et al. (2013) apresentam solução algorítmica para análise de logs de partidas de futebol de robôs simulados e extração de *setplays* a partir de sequencias de eventos relevantes que resultam num gol. Desta forma, os agentes podem aprender um novo *setplay* a partir de uma sequência de comportamentos não planejada de forma colaborativa que emerge a partir da interação natural dos agentes no SMA. Um novo plano passa a ser incorporado à biblioteca de *setplays* do time, potencializando a repetição daquelas jogadas que surgiram naturalmente e deram resultados positivos. A solução proposta pelos autores foi testada e validada no ambiente padrão da RoboCup Soccer Simulation 2D. A proposta também não utiliza conhecimento de um especialista humano, apenas generaliza num *setplay* jogadas que resultaram em gol a partir de comportamento cooperativo emergente do SMA.

Freelan et al. (2014) propõem uma abordagem promissora baseada em LfD. Consiste na decomposição de tarefas com objetivo de dividir os comportamentos complexos em tarefas menores e consequentemente reduzir sua dimensão. Faz uso de um sistema multiagente de aprendizagem supervisionada chamado *Hierarchical Training of Agent Behaviors* (HiTAB) que é usado para treinar os comportamentos de times de agentes colaborativos na forma de autômato finito determinístico (AFD) hierárquicos representados por uma máquina de Moore. Cada estado dos AFD representam um comportamento que o robô deve executar. De acordo com a diretriz de LfD, o objetivo é ensinar comportamento colaborativo a um time de robôs sem ter que programá-los, apenas demonstrando no campo de jogo o que se espera deles com poucos exemplos de treinamento online. A aprendizagem concentra-se em aprender as funções de transição nas máquinas de estado nos diversos níveis hierárquicos. Cada nível hierárquico representa uma camada de abstração. Quanto mais alto o nível na árvore de hierarquia, mais abstrato será o AFD. Quanto mais baixo, mais próximo de comportamentos concretos (ex: mover uma articulação, chutar, levantar, etc) estará o AFD. O algoritmo aprende exemplos de transições a partir de um demonstrador. Estes exemplos são generalizados através de um classificador para cada estado. Os autores utilizaram árvores de decisão (C4.5) para esta classificação. A abordagem foi aplicada para treinar o comportamento individual de robôs validado durante as competições RoboCup em 2011 e 2012. Um robô teve seu comportamento completamente treinado por demonstração poucos dias antes da competição e obteve resultado satisfatório, sendo o primeiro caso de robô com comportamento completamente aprendido por demonstração, sem comportamentos codificados manualmente.

Especificamente, o comportamento considerado por Freelan et al. (2014) era simples: o robô aprendeu a procurar a bola, ir até a bola, alinhar para o gol adversário, alinhar para chutar e chutar. Isto foi decomposto como uma hierarquia de autômatos e treinado utilizando o HiTAB. Os autores descrevem experimentos com aprendizagem de compor-

tamentos colaborativos. Para isto, utilizaram como base um código aberto da equipe UPennalizer de 2013[1]. A jogada consiste num *setplay* de passe onde um robô vai até a bola enquanto outro robô desloca-se para uma posição específica. Quando os dois robôs estão prontos, um primeiro robô chuta na direção do segundo e desloca-se para uma outra posição à frente. Então o segundo robô irá chutar na direção do primeiro robô e este, por fim, chutará para o gol. Para treinar esta jogada, cada robô foi treinado com o HiTAB individualmente utilizando agentes fantasma (*dummy*) como parceiro. Um agente fantasma pode ser implementado como um robô fictício inserido no modelo de mundo dos agentes que controlam os robôs reais. O agente fantasma também pode ser materializado como um robô desligado, que é colocado em campo em posições específicas, sem estar animado pelo controle de um agente inteligente. É um método muito utilizado para o treinamento de times de robôs jogadores de futebol.

Ao término do experimento, os dois conjuntos de autômatos aprendidos por cada robô separadamente foram unidos. A união dos comportamentos aprendidos resultou numa jogada colaborativa bem sucedida. Os autores consideram que para quantidade maior de agentes envolvidos ou *setplays* mais complexos esta abordagem de aprendizagem individual de cada agente não é suficiente e a interação precisa ser considerada durante o processo de aprendizagem.

Apesar de ser uma proposta bastante promissora que utiliza o conhecimento do especialista humano para aprender desde os comportamentos básicos até jogadas colaborativas de alto nível, Freelan et al. (2014) avaliaram apenas comportamentos de baixo nível como chute de um atacante em time de robôs humanoides. A jogada ensaiada que foi treinada não envolve comunicação, nem análise de variações dinâmicas no ambiente (ex: adversários roubar a bola, passe falhar, etc). Não está claro como os agentes podem aprender *setplays* complexos, transições, condições de início e término, além de adaptação às situações dinâmicas de jogo. A abordagem também não representa informações incertas, imprecisas ou incompletas, comuns no domínio de futebol robótico, no modelo de AFD utilizado.

Por outro lado, Fabro, Reis e Lau (2014) apresentam uma solução para aprender a escolha da melhor ação dentro de um *setplay* que tenha múltiplos caminhos. Desta forma, num *setplay* que possua transições distintas a partir do estado atual para diferentes estados sucessivos, a decisão de qual caminho seguir para cada situação dinâmica seria aprendida através do algoritmo *Q-Learning* multiagentes. Apesar da grande utilidade e da melhoria da adaptabilidade dos *setplays* existentes, esta abordagem não o aprende novos planos completos com todas as suas características. Poderá ser de grande valia para otimização dos planos aprendidos por demonstração, conforme proposta apresentada neste projeto.

Moradi, Ardestani e Moradi (2016) descrevem um framework para que especialistas humanos treinem comportamentos individuais do jogador que tem a posse de bola, em situações apresentadas ou criadas na interface gráfica (*play ground*). Utiliza a opinião ou julgamento de vários especialistas para cada solução apresentada para estabelecer o valor ou qualidade de cada proposta antes de ser treinada com o time de robôs. Segue a

[1]https://github.com/UPenn-RoboCup/UPennalizers

abordagem de LfD com o diferencial de trabalhar com grande quantidade de especialistas humanos. Os autores denominam esta característica de *crowdsourcing*. O treinamento dos robôs é feito utilizando aprendizagem por reforço. Apesar de seguir a mesma diretriz de LfD, aprende apenas comportamentos individuais para as situações apresentadas. Não há aprendizagem de cooperação nem o envolvimento dos robôs que não possuem a bola. O trabalho está em desenvolvimento e não foi realizado ainda experimentos conclusivos, apenas pequenas experiências iniciais demonstrando o *gap* entre as decisões humanas e dos robôs.

Shi et al. (2018) apresentam uma solução completa para tomada de decisão em um ambiente de futebol de robôs. A solução busca agregar as instâncias do *dataset* de treinamento e selecionar a estratégia adequada a partir das informações do ambiente, ainda que incompletas ou imprecisas. Utilizam três métodos para o processo de aprendizagem: *Support Vector Machine* (SVM), árvores de decisão e aprendizagem por reforço. Mesmo aprendendo o procedimento completo de tomada de decisão, não há a aprendizagem de planos cooperativos mas de comportamentos individuais que possam favorecer a um desempenho global eficaz do SMA. Nesta abordagem, não são aprendidos *setplays* de forma coordenada, mas sim políticas de ação individuais que geram um melhor desempenho global ao time. O objetivo é semelhante ao do *Situation Based Strategic Positioning* (SBSP)(REIS; LAU; OLIVEIRA, 2001) . A diferença é que Shi et al. (2018) utilizam a aprendizagem para definir de forma mais adaptativa o posicionamento, a partir dos exemplos de treinamento utilizados. Esta proposta abre possibilidade inclusive para posicionamentos estratégicos que favorecem mais os comportamentos ativos dos demais agentes.

Bianchi et al. (2018) propõem solução para transferir conhecimento aprendido entre diferentes domínios de problema, utilizando *Case-Based Reasoning* (CBR) . Por exemplo, transferir conhecimento do ambiente RoboCup Soccer Simulation 2D para robôs físicos similares aos simulados. Os autores apresentam experimentos bem sucedidos com dois pares de domínios. Apesar de não representar um aprendizagem de *setplays*, é uma abordagem interessante para transferir *setplays* desenvolvidos ou aprendidos para outros domínios, por exemplo, do ambiente RoboCup Soccer Simulation 3D para RoboCup Humanoid ou RoboCup Standard Platform League. Apesar deste benefício, não trata especificamente do aprendizagem de *setplays* ou planos cooperativos.

Destaca-se também na literatura o uso de aprendizagem de máquina profunda no domínio de futebol robótico. Esta abordagem tem sido largamente utilizada com algoritmos de aprendizagem por reforço para criação de novas habilidades em robôs humanoides (ABREU et al., 2019). Estes autores utilizam o algoritmo *Proximal Policy Optimization* (PPO) para otimizar movimentos como corrida e condução de bola a partir dos modelos existentes. Os resultados obtidos foram validados com robôs simulados no ambiente *RoboCup 3D Soccer Simulation* (3DSSIM) com desempenho melhor que a maioria dos grupos de pesquisa que participam desta competição RoboCup. Entretanto, alguns movimentos ainda não eram adequados para o uso competitivo ou para a finalidade principal a que se propõe em um robô humanoide jogador de futebol. A continuação deste trabalho apresenta um novo movimento de corrida muito rápido, pioneiro dentre todos os trabalhos já apresentados com corrida de robôs humanoides (ABREU; REIS; LAU,

2019). O mesmo algoritmo PPO foi usado desta vez com controle direto das articulações dos robôs. Os autores apresentam informações detalhadas sobre os espaços de observação e ação e como eles são obtidos. Melo, Melo e Maximo (2021) também utilizaram o PPO para aprender um comportamento de corrida com os mesmo tipo de robô humanoide dos trabalhos anteriores. Os autores conseguiram superar em 50% a velocidade dos movimentos de corrida no estado da arte.

Abreu et al. (2021) utilizam os algoritmos PPO e *Soft Actor Critic* para otimizar o movimento de chute omnidirecional. Duas abordagens são utilizadas: na primeira o modelo existente de chute tem seus parâmetros otimizados através da rede neural utilizada. Na segunda abordagem, a aprendizagem se dá com as ações de controle de baixo nível das articulações dos robôs. Já Spitznagel, Weiler e Dorer (2021) também utilizaram o PPO para aprendizagem dinâmica de chutes em robôs simulados no ambiente 3DSSIM. Neste trabalho, os autores fizeram uma análise detalhada dos hiper-parâmetros do algoritmo demonstrando como o ajuste cuidadoso destes parâmetros pode gerar um ganho considerável na qualidade das habilidades aprendidas. O chute com a bola em movimento também é alvo de investigação (TEIXEIRA et al., 2020). Nenhum destes trabalhos aplicou os métodos de aprendizagem por reforço profunda para aprender planos cooperativos no futebol de robôs. Estes trabalhos demonstram uma tendência clara ao uso da aprendizagem por reforço profunda, favorecida especialmente pelo advento do *Open AI Gym* (BROCKMAN et al., 2016). O *Gym* é um framework que torna mais rápido e ágil a criação de ambientes de treinamento para aprendizagem por reforço, além de contar com biblioteca com os mais populares algoritmos desta categoria. Entretanto, podemos observar que todos estes trabalhos concentram-se na aprendizagem de habilidades individuais dos robôs como caminhada, corrida, chute, etc. Nenhum destes trabalhos tem por objetivo a aprendizagem de estratégias coletivas ou planos coordenados no SMA.

SILVA, COSTA e STONE (2019) apresentaram resultados iniciais de um promissor trabalho em andamento. Eles usaram uma abordagem de aprendizagem por reforço distribucional comparando-a com o algoritmo DQN. Neste proposta, o retorno do algoritmo de aprendizagem é uma distribuição probabilística do valor esperado de cada ação, ao invés da média como nos algoritmos clássicos de aprendizagem por reforço baseados na aprendizagem Q. A validação foi realizada em um desafio comum utilizado pela comunidade *RoboCup 2D Soccer Simulation* (2DSSIM) conhecido como Meio-campo ofensivo. Consiste em um episodio em metade do campo de futebol em que um time ataca e outro defende. Os resultados mostraram-se promissores, mas não demonstram ainda a capacidade aprender planos coordenados completos (setplays) ou mesmo de capturar o conhecimento dos humanos no processo.

Nesta mesma linha, Ocana et al. (2019) utilizaram o simulador B-Human Framework (RöFER et al.,) que simula jogos de futebol entre robôs humanoides. O trabalho pretende que os robôs, jogando em duplas, aprendam jogadas coletivas a partir da aprendizagem por reforço. O ponto principal deste trabalho e a comparação entre a aprendizagem individual - quando cada agente irá aprender ações individuais - com a aprendizagem em conjunto, em que as ações de cada agente é combinada para gerar recompensas coletivas para o time. Nas duas abordagens, os autores utilizaram o algoritmo DDPG. Todos estes trabalhos indicam a tendência no estado da arte ao uso da aprendizagem de máquina

para aprender comportamentos coletivos emergentes. Não foi encontrado qualquer publicação que tenha apresentado resultados na aprendizagem de uso de planos coordenados recomendados por humanos.

Como pode ser observado, poucos trabalhos indicam uma tendência ao desenvolvimento das abordagens baseadas em LfD para aproveitar a visão crítica dos observadores humanos para apontar oportunidades de melhor desempenho dos agentes num SMA. O uso de LfD leva a uma aumento possivelmente exponencial no tamanho dos *datasets*, especialmente no que se refere à quantidade de instâncias. Muitos especialistas diferentes podem ser consultados e provocados a gerar suas opiniões, por exemplo, sobre jogadas em partidas de futebol de robôs. Ao considerar como especialistas treinadores de futebol, jogadores, fãs, jornalistas, chega-se a uma quantidade de pessoas, na faixa de milhões, ao redor do mundo que estariam aptas a atuar como especialistas indicando melhores jogadas aos robôs. Naturalmente, muitas jogadas, apesar de possuírem dados com valores ligeiramente distintos, representarão semanticamente um mesmo *setplay*. Um desafio aberto é identificar instâncias distintas, mas semanticamente equivalentes, em um *dataset* grande, com muitas instâncias.

Na próxima seção, trabalhos que tratam do tema de agrupamento de instâncias semanticamente equivalentes em *datasets* são destacados.

2.3 AGRUPAMENTO DE INSTÂNCIAS SEMANTICAMENTE EQUIVALENTES EM DATASETS

Para resolver o problema de instâncias em um dataset que têm o mesmo significado semântico para o senso comum no domínio do problema, é necessário organizar o dataset em grupos contendo instâncias equivalentes. Esta seção apresenta os trabalhos relacionados à organização de datasets em grupos, buscando identificar eventuais lacunas em relação ao agrupamento por equivalência semântica.

Yang et al. (2012) apresentam solução, baseada em algoritmos genéticos, para avaliação da similaridade das instâncias no agrupamento com *Fuzzy C-Means* (FCM). Os autores validaram sua proposta em *datasets* de *benchmark* populares na comunidade de Agrupamento e Mineração de Dados, mas não realizaram validação utilizando *datasets* reais na área de planejamento em SMA ou *setplays*. Não está claro se a abordagem proposta por Yang et al. (2012) reflete a equivalência semântica das instâncias ou apenas busca reduzir o tamanho do *dataset*.

Shao, Shi e Yu (2013) descrevem solução de agrupamento para lidar com múltiplos *datasets* incompletos. Assim, exemplos com propriedades não informadas em diversos *datasets* seriam identificados por semelhança. A abordagem inspira parcialmente a solução apresentada neste trabalho para tratar a a equivalência semântica, conforme descrito no capítulo 4. Entretanto, não trabalha especificamente com *datasets* similares aos de planejamento em SMA ou *setplays*. Também não trata equivalência semântica de instâncias e sim incompletude de propriedades (*features*).

Ziani (2014) utiliza seleção de *features* que representam objetos simbólicos. Um objeto simbólico pode representar uma categoria de objetos, podendo ser a base para um sistema de equivalência semântica como o que apresentamos posteriormente no capítulo

4. Entretanto, os autores não mencionam claramente o uso da solução para representar equivalência semântica entre instâncias num *dataset* qualquer e em especial relativo a planejamento em SMA ou *setplays*. Deixam, portanto, uma lacuna para investigação da aplicabilidade desta abordagem para este fim.

Hieu e Meesad (2015a) realizam agrupamento de *datasets* grandes de forma a adaptá-los ao uso em computadores com memória reduzida. Propõem soluções para determinação do parâmetro K (quantidade de grupos) e dos centroides e mapeamento do *grid* no algoritmo *K-means* de acordo com a memória disponível. É uma solução indicada para *datasets* com baixa dimensionalidade, mas com alto número de instâncias, semelhantes ao que se espera obter como resultado do uso de LfD para aprendizagem de *setplays*. Apesar de considerar a similaridade das instâncias no agrupamento, não está aplicado especificamente a *datasets* para planejamento de ações em um SMA. As contribuições podem ser úteis para os propósitos deste projeto, mas precisariam ser validadas e/ou adaptadas para a realidade de aprendizagem de planos multiagentes. Um ponto importante a ser considerado é o nível de similaridade entre as instâncias agrupadas. Os agrupamentos gerados precisariam refletir um mesmo *setplay* ou uma mesma condição de ativação de *setplays*.

Hieu e Meesad (2015b) apresentam um método capaz de reduzir em 30% o tempo de execução do *K-Means* em *datasets* grandes. A abordagem baseia-se na melhoria do *K-Means* Paralelo com *MapReduce*, mantendo uma acurácia de 98%. Trata-se de mais um trabalho sobre agrupamento em *datasets* que não incluem *datasets* com foco em planejamento de SMA. Não está claro o quão bom é o agrupamento para saber se seria possível identificar semanticamente o mesmo *setplay* representado por várias instâncias. Precisaria ser investigada a viabilidade do uso desta abordagem para identificação da equivalência semântica entre duas ou mais instâncias que representam um mesmo *setplay*.

Wójcik, Zdunek e Antończak (2016) demonstram a possibilidade de usar métodos não supervisionados para verificação dos parâmetros de entrada de algoritmos de agrupamento. Utiliza *K-Means* aplicado a *dataset* de espectros obtidos por um equipamento laser para indução da degradação de materiais (LIBS). Apesar de demonstrar o agrupamento de instâncias similares de um *dataset* e apresentar solução para verificação dos parâmetros de entrada do *K-Means*, não é aplicado a comportamentos ativos como o planejamento multiagentes e sim dados estáticos obtidos a partir de um equipamento experimental de laboratório. O *dataset* e domínio do problema utilizados têm natureza muito distinta do que se espera obter como resultado de um processo de treinamento numa abordagem de LfD. Seria necessário investigar a aplicabilidade desta solução para o contexto do presente projeto.

Xu, Li e Zhong (2017) apresentam solução para re-classificação de instâncias nas bordas de *clusters* visando minimizar os erros de classificação comuns nestas instâncias. O foco é lidar com *datasets* incompletos. Utiliza o método FCM para definição dos *clusters*. O objetivo central é minimizar os erros de classificação, mas não há garantias de obter a equivalência semântica necessária para o objeto de estudo deste projeto. O trabalho lida especialmente com instâncias incompletas, com propriedades não preenchidas.

Sardar, Ansari e Khatun (2017) propõem versão modificada do *K-means* para executar segundo o paradigma *MapReduce* de processamento distribuído para lidar com *datasets*

grandes. Dentre as modificações propostas, não há nenhuma que vise considerar a equivalência semântica entre duas instâncias como critério para inclusão no mesmo grupo. Sardar, Ansari e Khatun (2017) não utilizaram *datasets* de planejamento em SMA ou *setplays* em seus experimentos.

Não foram encontrados trabalhos que tratem a organização do dataset, considerando a equivalência semântica entre as instâncias, representando uma clara lacuna no estado da arte.

2.4 CONSIDERAÇÕES FINAIS

Ao final da revisão de literatura, foi possível responder às questões de pesquisa propostas no início deste capítulo. A primeira questão refere-se aos métodos de aprendizagem de máquina utilizados em planejamento em SMA. Neste sentido, foi observada uma boa variedade de métodos com algum destaque para aprendizagem por reforço, seguido de redes bayesianas. Diversos trabalhos apresentados confirmam esta tendência com especial ênfase ao uso do ambiente Open AI Gym e a aprendizagem profunda.

Na segunda questão, foram observadas as abordagens que utilizam LfD. As duas principais abordagens encontradas referem-se a decomposição de tarefas (HiTAB) e *crowdsourcing*.

Este cenário indica que há tentativas recentes de utilizar aprendizagem de máquina para permitir que times de agentes aprendam planos colaborativos. No entanto, não foram encontrados trabalhos que conseguem aprender planos completos com todos os estados, comportamentos de todos os agentes em cada estado, condições de início, finalização, cancelamento e transições. Adicionalmente, há algumas iniciativas em direção à tentativa de utilizar o conhecimento informal do especialista humano através das abordagens de LfD. Em especial num domínio com uma grande quantidade de especialistas disponíveis, como o futebol, esta abordagem apresenta-se promissora. Também não foram encontrados trabalhos que tenham realizado experimentos com aprendizagem de *setplays* complexos em todas as suas características utilizando LfD.

A terceira questão foi levantada por uma consequência direta da hipótese de usar LfD num domínio em que as fontes de dados de treinamentos potencialmente podem ser geradas por uma grande quantidade de especialistas. A implicação imediata desta abordagem é a geração de muitos *datasets* com grande quantidade de instâncias, mesmo que com baixa dimensionalidade. Ademais, a probabilidade do mesmo *setplay* ser recomendado por especialistas humanos diferentes em situações idênticas é alta. O mesmo especialista pode indicar um mesmo *setplay* para situações diferentes. Estas duas possibilidades irão gerar instâncias diferentes num *dataset* mas que têm o mesmo valor semântico. Num processo de treinamento a partir deste *dataset* é importante identificar a equivalência semântica entre estas instâncias antes de realizar o treinamento. Foram investigadas, portanto, as soluções existentes quanto a agrupamento de instâncias com equivalência semântica em *datasets* grandes. Os principais trabalhos encontrados utilizam *K-Means* ou FCM. Não foram encontrados trabalhos que tenham provado detectar instâncias semanticamente equivalentes em *datasets* relacionados ao planejamento em SMA ou *setplays*. Há então uma clara lacuna a ser investigada quanto à viabilidade de usar os métodos en-

contrados neste domínio ou a necessidade de adaptá-los ou criar novas soluções capazes de atingir os objetivos aqui descritos.

Poucos trabalhos avaliados levam em consideração uma necessidade básica de ambientes similares ao futebol de robôs: a representação de conhecimento incerto, impreciso ou incompleto. Muitas das decisões num *setplay* dependem de dados imprecisos como, por exemplo, distâncias. Ainda como exemplo, considere uma situação de comando para um determinado robô: "Se seu oponente estiver muito perto, faça um passe para um companheiro de time". O conceito de **muito perto** não é preciso, não há um "número mágico"que defina a partir de que distância considera-se muito perto. Ele é variável e depende de muitos fatores dinâmicos. Distâncias em relação a outros robôs, à bola, ao gol adversário e do próprio time entre outros são exemplos de grandezas cujos valores são naturalmente nebulosos no processo de tomada de decisão. Ademais, considerando robôs com sistema de visão parcial, em muitos instantes os agentes não terão o conhecimento completo do mundo, ou terão conhecimento atrasado (referente a ciclos de visão mais antigos) pois nem sempre todos os objetos estão em seu campo de visão. Pelo exposto, faz-se necessário tratar este tipo de informação imprecisa ou incompleta.

A quarta questão busca identificar soluções para lidar com este problema. Observou-se que alguns poucos trabalhos utilizam redes bayesianas para representar as relações de interdependência do comportamento dos agentes no planejamento em um SMA. Os trabalhos que utilizam lógica fuzzy estão relacionados sempre às soluções de agrupamento de instâncias em *datasets*, não encontrando nenhum trabalho recente relacionado a aprendizagem de *setplays* utilizando lógica fuzzy. Nesta questão evidencia-se talvez uma das maiores lacunas encontradas no estado da arte: a necessidade de atender a um requisito natural do domínio do problema que é representar os dados incertos, imprecisos ou incompletos durante o processo de aprendizagem de planos multiagentes ou *setplays*.

Respondidas as quatro questões de pesquisa, considera-se o objetivo da revisão de literatura atingido. Levantou-se os principais métodos utilizados na classe de problemas considerada e identificou-se as diversas lacunas a serem investigadas no estado da arte.

Considerando estas informações, elaborou-se uma proposta de solução capaz de aprender *setplays* completos no domínio do futebol robótico, utilizando o conhecimento informal de espectadores humanos a partir do paradigma LfD, tratando as questões de incerteza e similaridade semântica entre as instâncias do *dataset* de treinamento. Esta proposta diferencia este trabalho dos demais encontrados no estado da arte. A proposta está detalhada no capítulo 4. Além dos resultados apresentados, este trabalho tem como contribuições adicionais:

- uma nova versão da ferramenta *Strategy Planner* (SPlanner)(CRAVO et al., 2014) com novos comportamentos adequados ao ambiente de testes definido pela 3DSSIM e importar situações diretamente das partidas gravadas;

- uma nova versão do software de visualização RoboViz usado no ambiente 3DSSIM para permitir ao especialista extrair a situação de jogo para o qual deseja enviar uma demonstraçao;

- um *dataset* de *setplays* recomendados por especialistas em futebol e robótica, cole-

tado através de uma estratégia de *crowdsourcing*

- uma solução da agrupamento fuzzy capaz de organizar o dataset de demonstrações, considerando a equivalência semântica
- um mecanismo de aprendizagem por reforço, capaz de aprender uma política de seleção de *cluster* no *dataset* para usar em cada situação de jogo.

O resultado desta tese permite que a política de uso dos planos recomendados pelos especialistas adapte-se às habilidades dos robôs de cada equipe, permitindo o uso mais abrangente do *dataset* formado durante este trabalho.

No próximo capítulo serão apresentados os fundamentos teóricos que embasam o desenvolvimento deste trabalho com destaque para a descrição do ambiente de experimentação da liga 3DSSIM da RoboCup, métodos de agrupamento considerando incerteza para lidar com a equivalência semântica e métodos de aprendizagem por reforço para aprender a política de controle para escolha dos *setplays* para uso em cada situação.

Capítulo 3

Neste capítulo, apresentam-se os principais conceitos teóricos utilizados neste trabalho, incluindo: sistemas multiagentes, o desafio RoboCup de futebol de robôs simulados, setplays, agrupamento fuzzy e aprendizagem por reforço.

FUNDAMENTAÇÃO TEÓRICA

Este trabalho possui três pilares teóricos: sistemas multiagentes, agrupamento fuzzy e aprendizagem por reforço. O problema investigado está no contexto da inteligência artificial coletiva. Apresentam-se evidências da capacidade de artefatos inteligentes serem treinados a partir do conhecimento intuitivo humano coletado massivamente na internet. Em especial, este trabalho concentra-se nos Sistemas Multiagentes (SMA) descritos na seção 3.1.

Para organizar o *dataset* coletado, utilizou-se uma abordagem hierárquica de agrupamento *fuzzy*. Os fundamentos teóricos desta abordagem são descritos na seção 3.2. Por fim, o treinamento do SMA utiliza os fundamentos da aprendizagem por reforço apresentados na seção 3.3. Neste capítulo, apenas algoritmos e ferramentas utilizados neste trabalho são abordados.

3.1 O DESAFIO DE FUTEBOL DE ROBÔS SIMULADOS ROBOCUP

Um SMA é um sistema em que vários agentes inteligentes interagem entre si enquanto buscam atingir um conjunto de objetivos ou realizar um conjunto de tarefas(WEISS, 1999). Em muitos casos, os agentes num SMA agem em nome de usuários ou seus proprietários com objetivos e motivações diferentes. Portanto, os agentes precisam ter a habilidade de cooperar, coordenar e negociar entre si, da mesma forma como os humanos o fazem em suas atividades diárias(WOOLDRIDGE, 2009).

Os domínios de problema solucionados com o uso de SMA podem apresentar diferentes níveis de dificuldade. A modelagem do problema para os agentes é descrita através do ambiente em que os agente atuam. Algumas propriedades dos ambientes destes agentes costumam ser avaliadas para definir seu nível de dificuldade(RUSSELL; NORVIG, 2021):

- completamente ou parcialmente observável: no ambiente completamente observável os agentes sempre conseguem obter a qualquer instante todas as informações relevantes através dos seus sensores. No ambiente parcialmente observável, apenas

parte das informações do ambiente estão disponíveis ou as informações não estão disponíveis todo o tempo.

- determinístico ou estocástico: em um ambiente determinístico, o próximo estado do ambiente pode ser completamente determinado a partir do estado atual e da ação escolhida pelo agente para executar. Assim, o agente consegue prever o efeito imediato de suas ações com precisão. Os ambientes que não possuem esta propriedade são chamados estocásticos.

- episódico ou sequencial: em um ambiente episódico, a experiência dos agentes é dividida em episódios atômicos. Em cada episódio, cada agente recebe percepções através dos sensores, seleciona uma ação e executa-a. O próximo episódio não depende das ações executadas no episódio anterior. Nos ambientes sequenciais, as decisões tomadas no episódio atual podem impactar as decisões que serão tomadas em episódios futuros. As ações imediatas podem ter consequências a longo prazo.

- estático ou dinâmico: se o ambiente e suas características podem mudar enquanto os agentes decidem qual ação irão executar, o ambiente é dinâmico. Neste caso, o tempo de deliberação é fundamental pois quando a ação escolhida for executada o ambiente pode estar bem diferente do que aquele indicado nas ultimas percepções capturadas pelo agente. Quando não há possibilidade do ambiente ter seu estado modificado durante a deliberação dos agentes, o ambiente é estático.

- discreto ou contínuo: a distinção entre discreto e contínuo aplica-se aos estados possíveis de um ambiente, à maneira como o tempo é considerado e às percepções e ações. Quando qualquer uma destas características é contínua, o ambiente é considerado contínuo. Se o espaço de estados, o tempo, as percepções e ações são todos conjuntos discretos, o ambiente é considerado discreto.

Destas propriedades, pode-se inferir que o ambiente mais desafiador para um SMA é parcialmente observável, estocástico, sequencial, dinâmico e contínuo. Desde os anos 90, um desafio padrão foi proposto para representar esta classe de problemas: o futebol de robôs. Considerando um time de 11 robôs humanoides disputando uma partida de futebol contra outro time de 11 jogadores (humanos ou robôs), o ambiente é parcialmente observável(a visão de cada robô é dada por uma câmera embarcada em sua cabeça com um ângulo de visão limitado), estocástico(em função da ação dos outros jogadores, da imprecisão dos sensores e atuadores, não há como prever de maneira determinística o próximo estado a partir do estado atual e da ação selecionada), sequencial (as jogadas num jogo de futebol são um encadeamento de ações coordenadas entre os jogadores, por isto há dependência entre as decisões em cada episódio), dinâmico(o ambiente continua mudando continuamente, por exemplo, a bola continua movendo-se, os outros jogadores também continuam movendo-se enquanto o agente delibera) e contínuo (tanto o espaço de estados, quanto as ações e percepções são contínuas). Desta forma, este é um bom desafio padrão para ser utilizado em projetos científicos que buscam avançar o estado da arte em SMA.

Em 1997, foi criada a iniciativa RoboCup. A Copa do Mundo de Robôs, como ficou conhecida, propõe um desafio científico para a Inteligência Artificial (IA) e a Robótica utilizando inicialmente o futebol de robôs como problema padrão. Embora seja tão óbvio que construir um robô para jogar futebol seja um desafio imenso, a motivação dos idealizadores da RoboCup não se resumia a isto. A intenção foi criar um veículo para revitalizar a pesquisa de IA, oferecendo um desafio publicamente atraente, mas formidável. Uma das maneiras eficazes de promover a pesquisa em computação e engenharia é estabelecer uma meta significativa de longo prazo. Quando a realização de tal objetivo tem impacto social significativo, é chamado de projeto do grande desafio (KITANO et al., 1998). Porém, construir um robô para jogar futebol por si só não gera impacto social e econômico significativo, mas a realização deste objetivo certamente será considerada como uma grande conquista da área. Chama-se esse tipo de projeto de projeto de referência. A RoboCup é um projeto de referência, bem como um problema padrão (KITANO et al., 1998). Um problema padrão caracteriza-se por ter regras bem definidas e universalmente conhecidas, além de ser aceito pela comunidade científica como representante de uma classe de problemas investigado em algum domínio específico.

A competição mundial RoboCup continua sendo realizada anualmente pela RoboCup Federation[1], uma sociedade científica sem fins lucrativos criada para fomentar a pesquisa científica em IA e Robótica. A iniciativa envolve, anualmente, pesquisadores e estudantes de mais de 50 países. Em alguns países, sociedades científicas locais organizam competições nacionais, a exemplo da Competição Brasileira de Robótica organizada pela RoboCup Brasil[2]. A iniciativa RoboCup contempla diversos desafios e problemas padrão, mantendo o futebol de robôs como o seu grande desafio. Para melhor organização das competições, os desafios são divididos em ligas. O Futebol de Robôs é subdividido em cinco ligas: *Humanoid*, *Standard Platform*, *Middle Size*, *Small Size*, *Simulation*. A liga de Futebol Simulado(*Simulation*) é a única que prevê em suas regras jogos entre times formados por 11 robôs, sendo a mais interessante para estudar os problemas inerentes aos SMA, devido à complexidade do ambiente.

A liga de Futebol Simulado é subdivida em Simulação 2D (Sim-2D) e Simulação 3D (Sim-3D). A Sim-2D é uma das primeiras competições da RoboCup, criada em 1997 no primeiro ano da iniciativa. A Sim-2D simula um mundo bidimensional com robôs cilíndricos utilizam ações abstratas como Girar, Chutar, Correr e Pegar. O simulador possui modelo de ruído, sensores e atuadores assíncronos, tomada de decisão em tempo real, visão e audição restritos e modelo de resistência do jogador. Resumindo, a Sim-2D oferece um grande leque de características realistas em sua simulação. Entretanto, falta-lhe a terceira dimensão. O mundo real é tridimensional. Para suprir esta lacuna, em 2004 surgiu a Sim-3D. Inicialmente adotaram-se agentes na forma de esferas com ações abstratas similares à Sim-2D. Mas logo ficou claro que, apesar de interessante, este modelo não contribuía muito com o objetivo de longo prazo da RoboCup de fazer robôs humanoides jogarem futebol. Então, em 2007, a Sim-3D sofreu uma grande mudança adotando o *Soccerbot*: um modelo humanoide baseado no robô HOAP-2 da Fujitsu(ver

[1]https://www.robocup.org/
[2]https://www.robocup.org.br/

Figura 3.1)(KALYANAKRISHNAN et al., 2010).

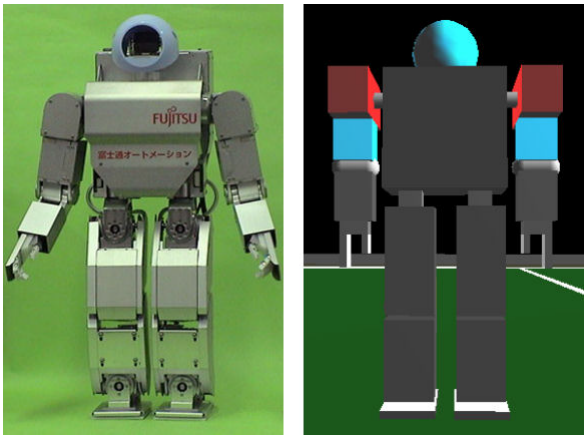


Figura 3.1: Robô HOAP-2: (a) versão real (b) versão da Sim-3D

Pela primeira vez, na liga de Simulação, os robôs precisavam ser programados utilizando uma interface de baixo nível controlando o torque dos motores. Apesar da queda inicial na qualidade dos jogos, os benefícios de longo prazo fizeram valer à pena a mudança. Em 2008, um novo modelo humanoide com maior quantidade de graus de liberdade foi incluído na Sim-3D substituindo o HOAP-2: o robô NAO da Softbank Robotics[3](KALYANAKRISHNAN et al., 2010). A Figura 3.2 ilustra as versões real e simulada do NAO. Além do maior número de motores e articulações, o robô aproxima-se mais dos modelos reais utilizados nas ligas físicas de futebol da RoboCup. Este modelo continua sendo utilizado atualmente, incorporando agora algumas variações que permitem o uso de robôs heterogêneos. Algumas variações incluem articulações extras com dedos nos pés enquanto outras possuem motores com velocidades maiores ou pernas mais longas, por exemplo.

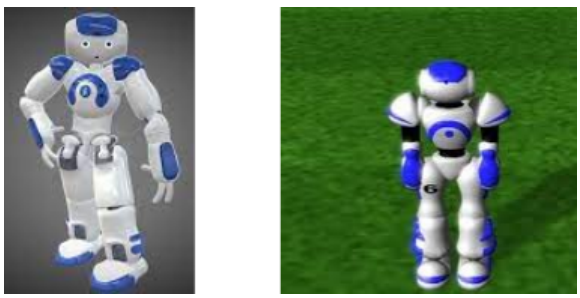


Figura 3.2: Robô NAO: (a) versão real (b) versão da Sim-3D

O simulador utilizado na Sim-3D é o SimSpark(BOEDECKER; ASADA, 2008) com o plugin RCSS3D para o ambiente do futebol de robôs. O pacote completo é licenciado

[3]https://www.softbankrobotics.com/emea/en/nao

como software livre desenvolvido pela comunidade RoboCup voluntariamente[4]. Além do simulador, o ambiente da Sim-3D é composto por outras duas ferramentas: o RoboViz[5] e o magma Proxy[6]. O RoboViz é uma ferramenta de visualização de jogos. Ele permite assistir às partidas de futebol de robôs na Sim-3D em tempo real ou reproduzidas a partir dos logs de simulação. É o visualizador utilizado nas competições oficiais. A Figura 3.3 exibe a tela do RoboViz.

Figura 3.3: Imagem do RoboViz: o visualizador de jogos oficial da Sim-3D.

O RCSS3D utiliza um modelo de tempo dividido em ciclos de simulação de $0,02$ segundos. Desta forma, o modelo de tempo que, originalmente, é contínuo, é transformado em discreto. A cada ciclo, todos os agentes recebem um conjunto de percepções do ambiente e têm a oportunidade de enviar um conjunto de ações para o simulador. Perder mensagens, sensoriais ou de atuação, pode gerar uma queda de desempenho dos agentes. Entretanto, mesmo que o agente tenha sido projetado com esta preocupação, estas perdas podem acontecer devido à fatores externos, como por exemplo a latência da rede de computadores. Para eliminar a influência dos fatores externos, foi desenvolvida uma camada de software chamada magmaProxy[7] que executa localmente no mesmo computador em que os agentes que controlam um time são executados. Esta camada garante a recepção das mensagens enviadas pelos agentes a cada ciclo, enfileirando-as e enviando ao simulador que, usualmente, executa em outro computador na rede. Desta forma, as questões de latência de rede são minimizadas garantindo um desempenho mais estável para os agentes. O magma Proxy faz parte do ambiente oficial de competições da Sim-3D. A Figura 3.4 ilustra o ambiente completo.

Neste ambiente, são utilizados quatro computadores. O computador Servidor é dedicado ao SimSpark e ao RCSS3D, executando a simulação completa do jogo. O Monitor

[4]Repositório do Simulador 3D: https://gitlab.com/robocup-sim/SimSpark
[5]https://github.com/magmaOffenburg/RoboViz
[6]https://github.com/magmaOffenburg/magmaProxy
[7]https://github.com/magmaOffenburg/magmaProxy

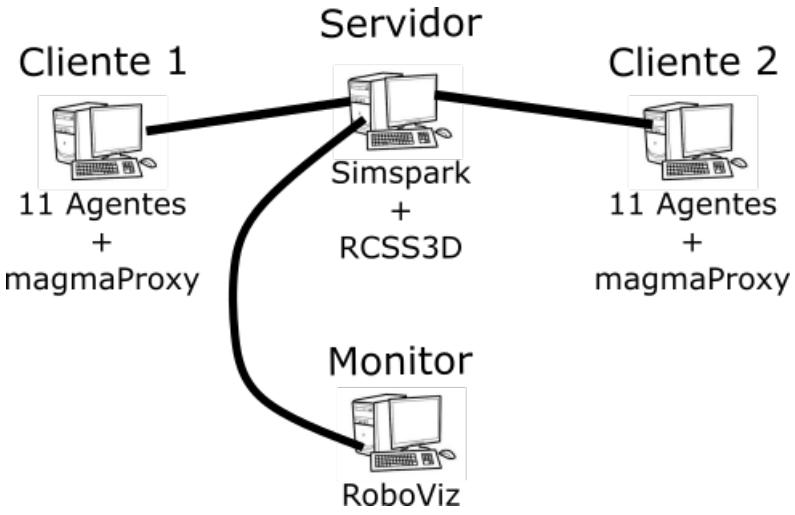


Figura 3.4: Ambiente oficial da Sim-3D

exibe a partida em tempo real através do RoboViz. É importante ter uma boa placa gráfica no Monitor para renderizar as imagens com melhor qualidade e desempenho. O Cliente 1 e Cliente 2 executam, cada um, 11 agentes segregados em processos independentes que não se comunicam e o magmaProxy. O magmaProxy será a camada intermediária entre os 11 agentes e o simulador. Os agentes conectam localmente no magmaProxy e este gerencia toda a comunicação de rede com o SimSpark. Toda a comunicação utiliza protocolo de rede TCP/IP.

Apesar da configuração descrita, é possível executar o ambiente utilizando configurações alternativas. Por exemplo, executar o RoboViz e um time em um computador e o simulador e outro time em um segundo computador utilizando somente duas máquinas. O ambiente da Figura 3.4 representa o ambiente recomendado e adotado nas competições oficiais. O uso de quantidade menor de computadores ou computadores com pouco poder de processamento ou pouca memória principal pode interferir no desempenho dos agentes nos jogos, pois eles podem perder muitos ciclos de troca de mensagens com o simulador, prejudicando suas ações básicas como caminhar, levantar e chutar.

O ambiente da Sim-3D é, portanto, um excelente cenário de testes para SMA e, em particular, para o propósito da aprendizagem de comportamentos colaborativos. Sendo assim, nesta tese, o ambiente da Sim-3D será utilizado sobre uma estratégia de *crowdsourcing* para construir um *dataset* grande com centenas de instâncias de *setplays*.

3.1.1 Setplays

Em um SMA, um agente sempre se depara com uma situação definida pelas propriedades extraídas do ambiente no estado atual ou em estados anteriores. Este conjunto de propriedades indica como o agente vê o mundo, que é chamado de estado do mundo ou estado mental. Assim, uma sequência de situações resume toda a vida do agente. Essas situações podem ser estratégicas ou ativas. Sempre que o agente posicionar-se visando

favorecer o comportamento coletivo da equipe, há uma situação estratégica acontecendo. Caso contrário, quando um agente realiza comportamentos ativos (por exemplo, passar uma bola, driblar um bola, interceptação de bola, etc.) que geram mudanças relevantes no estado de ambiente, uma situação ativa está acontecendo(REIS; LAU; OLIVEIRA, 2001).

Quando um agente detecta uma situação ativa, ele pode decidir realizar um único comportamento ativo ou pode optar por disparar uma instância de um plano cooperativo predefinido (*setplay*). Cada *setplay* tem uma condição inicial C_I que identifica as situações em que este setplay pode ser ativado(MOTA; LAU; REIS, 2010).

Um *setplay* pode ser definido como um plano flexível de várias etapas envolvendo um número variável de robôs(MOTA; REIS; LAU, 2011). Originalmente, *setplays* foram definidos como uma combinação de(STONE; VELOSO, 1999):

- uma condição de ativação que define um conjunto de estados do mundo em que o *setplay* será ativado;
- um conjunto de $\ddot{m}$ papéis de setplay $R_{sp} = \{spr_1, \ldots, spr_{\ddot{m}}\}$ que define os comportamentos a serem executados pelos robôs participando do *setplay*. Cada papel do setplay spr_i, $i = 1, \ldots, \ddot{m}$, inclui:
  - um comportamento a ser executado; e
  - uma condição final que indica um conjunto de estados do mundo em que o agente deve abandonar seu papel no setplay e retomar seu comportamento normal.

Esta definição limita o *setplay* a um único fluxo de passos. No início, os *setplays* eram usados em situações específicas, como jogadas de bola parada (por exemplo, escanteios, cobranças de laterais, etc.) no futebol de robôs. Em seguida, o conceito foi estendido para suportar múltiplos fluxos de passos e aplicações para vários cenários(MOTA; REIS; LAU, 2011).

Os papéis do *setplay* foram renomeados como etapas ou passos e seu conceito foi estendido (MOTA; REIS; LAU, 2011). Passos são blocos fundamentais usados para construir *setplays*. *Setplays* são construídos a partir de um conjunto $S = \{S_0, S_1, \ldots, S_{\ddot{m}-1}\}$ contendo $\ddot{m}$ passos. Um passo representa um estado durante a ativação do *setplay*. Agentes que participam de um *setplay* podem fazer parte de todos os seus passos ou apenas de alguns. Cada passo tem uma identificação numérica única. Um passo S_i ($i = 0, \ldots, \ddot{m} - 1$) é definido como(MOTA; REIS; LAU, 2011):

- um conjunto de comportamentos $B_i = \{b_{i1}, \ldots, b_{i\ddot{a}^i}\}$, onde $\ddot{a}^i$ é o número de agentes participantes no passo S_i;
- um conjunto de condições de transição $\tau_i = \{T_{i1}, \ldots, T_{i\kappa^i}\}$, onde κ^i é o número de transições no passo S_i.

Um *setplay* deve ter um nome ou identificação único e uma lista de referências aos agentes que participam de uma ativação específica do *setplay*. Essas referências podem estar relacionadas a agentes específicos ou passos na definição da estratégia da equipe(REIS; LAU; OLIVEIRA, 2001). Um conjunto de condições de cancelamento $C_{f1}, C_{f2}, \ldots, C_{f\zeta}$ definem os estados do mundo que levam o *setplay* a ter sua execução cancelada. Além disto, uma condição C_s define a transição para um estado terminal que indica a conclusão do *setplay* com sucesso, onde $\{C_s, C_{f1}, \ldots, C_{f\zeta}\} \subset \bigcup_{i=0}^{\ddot{m}-1} \tau_i$ e ζ é o número de condições de falha.

Um *setplay* pode ser representado como um autômato finito determinístico (AFD), conforme ilustrado na figura 3.5. Os estados representam os passos do *setplay* e as transições em $\tau_s, \forall s \in S$, são as transições entre estados.

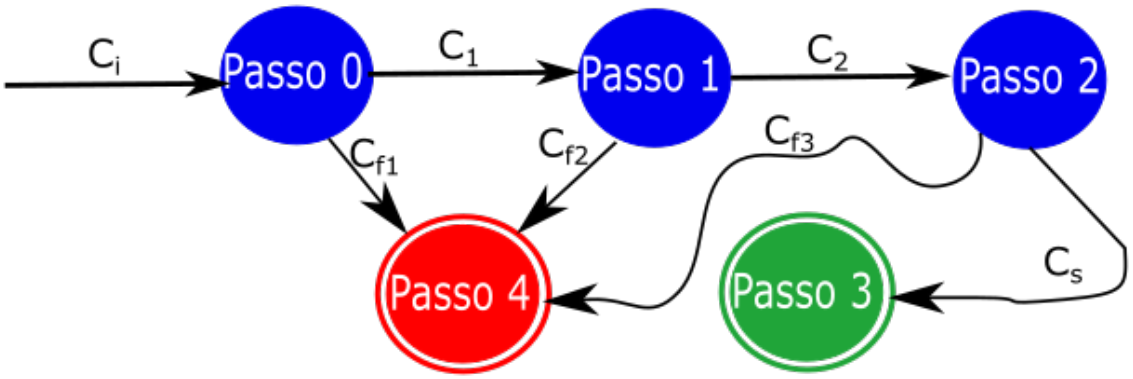


Figura 3.5: *Setplay* representado como um AFD. Os passos 0, 1 e 2 são os passos usuais do *setplay* incluindo seu passo inicial 0. O passo 3 é o passo final que indica o término com sucesso do *setplay*. O passo 4 indica que o *setplay* cancelou com falha. A condição C_I é a condição inicial que indica a situação em que o *setplay* é ativado. C_1 e C_2 são condições de transição e C_s, C_{f1}, C_{f2} e C_{f3} são condições terminais.

Um agente não muda seu comportamento em um mesmo passo. Então, podemos garantir que enquanto não houver uma transição para um novo passo, todos os agentes irão manter seus comportamentos. Para garantir que todos os agentes envolvidos no *setplay*, estejam sincronizados no mesmo passo, é essencial que haja uma coordenação entre os agentes utilizando uma política de sincronização e comunicação acordada previamente (MOTA; REIS; LAU, 2011). A política utilizada para os *setplays* define que, em cada passo, um agente é nomeado como líder. O líder é responsável por verificar se as condições de transição do passo atual são ou não satisfeitas. Quando alguma destas condições de transição forem satisfeitas, o líder informará através do protocolo e comunicação do SMA a mudança de passo. É preciso ter uma critério livre de ambiguidade para determinar quem é o líder em cada passo. No futebol de robôs, por exemplo, um critério amplamente utilizado é definir como líder o jogador que detém a posse de bola ou está mais próximo da bola.

Para organizar o *dataset* de *setplays* utilizou-se uma abordagem de agrupamento *fuzzy* que é o tema da próxima seção.

3.2 AGRUPAMENTO FUZZY

As abordagens de agrupamento são ferramentas construtivas para investigar estruturas de dados e surgiram como técnicas populares para reconhecimento não supervisionado de padrões. Essas técnicas são utilizadas em muitas áreas de aplicação, como reconhecimento de padrões, mineração de dados, aprendizado de máquina, etc(NAYAK; NAIK; BEHERA, 2015).

Uma abordagem de agrupamento, que envolve a minimização de alguma função objetivo, pertence ao grupo de algoritmos de função objetivo. Quando o algoritmo é capaz de minimizar o erro em uma função objetivo, é frequentemente chamado de c-Means sendo c o número de grupos ou *clusters*(NAYAK; NAIK; BEHERA, 2015). Se o processo de agrupamento utilizado utiliza a técnica *fuzzy* ou simplesmente difusa, então ele é conhecido como *Fuzzy C-Means* (FCM)(BEZDEK, 1981).

O agrupamento FCM usa uma associação difusa que atribui um grau de pertinência para cada objeto em todos os grupos. O benefício do FCM é a formação de novos *clusters* a partir dos pontos de dados que possuem valores de associação próximos aos grupos existentes. Basicamente, existem três operadores básicos no método FCM: a função de pertinência *fuzzy*, a matriz de partição e a função objetivo. No restante desta seção, apresentam-se estes conceitos e o algoritmo FCM clássico conforme descrito em Bezdek (1981).

Considere $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^p$ um conjunto de n objetos em um espaço p-dimensional. Uma partição *fuzzy* é uma família de c subconjuntos *fuzzy* de X ($2 \leq c \leq n-1$) representada por uma matriz, de ordem n x c, $U = [u_{ik}]$, onde $u_{ik} \in [0, 1]$ é o grau de pertinência de um objeto x_k ($1 \leq k \leq n$) no i-ésimo grupo ($1 \leq i \leq c$). Em uma pseudo-partição, para evitar que todos os objetos tenham seu maior grau de pertinência no mesmo grupo ou que todos eles tenham seu maior grau de pertinência em n subconjuntos diferentes de X, ou seja, cada objeto teria seu próprio grupo, uma partição *fuzzy* deve satisfazer as seguintes restrições

$$\sum_{i=1}^{c} u_{ik} = 1, \tag{3.1}$$

$$0 < \sum_{k=1}^{n} u_{ik} < n. \tag{3.2}$$

O algoritmo FCM associa o grau de pertinência de pertinência de um objeto x_k ao i-ésimo grupo através da seguinte equação

$$u_{ik} = \frac{d(x_k, v_i)^{\frac{-2}{m-1}}}{\sum_{i'=1}^{c} d(x_k, v_{i'})^{\frac{-2}{m-1}}}, \tag{3.3}$$

onde d é a função que mede a dissimilaridade entre os elementos no processo de agrupamento, m $(1 < m < \infty)$ é o expoente de ponderação *fuzzy*, chamado *fuzzificador*, que determina o nível de difusão dos graus de pertinência nos grupos, e $v_i \in V = \{v_1, \ldots, v_c\} \subset \mathbb{R}^p$ é o centróide do i-ésimo grupo definido como

$$v_i = \frac{\sum_{k=1}^{n} u_{ik}^m x_k}{\sum_{k=1}^{n} u_{ik}^m}, 1 \leq i \leq c. \tag{3.4}$$

A matriz de graus de pertinência U é chamada partição *fuzzy* ou pseudo-partição devido à sua entrada u_{ik} ser como a associação de x_k no i-ésimo subconjunto de particionamento *fuzzy* de X.O valor u_{ik} revela a proximidade de x_k em relação a v_i, ou seja, se x_k for mais próximo de v_i do que os outros centróides de grupos v_j $(1 \leq j \leq c, j \neq i)$, então u_ik será o máximo grau de pertinência de x_k.

O FCM é um processo iterativo que pode ser iniciado aleatoriamente com uma pseudo-partição $U^{(t)}$ ou por um conjunto de centróides $V^{(t)}$, onde t é o número da iteração atual definido inicialmente como $t = 0$. Os passos e iterações seguintes atualizam cada elemento de U pela equação (3.3) e cada elemento de V pela equação (3.4). Estas atualizações tentam minimizar a dissimilaridade entre um objeto x_k e um centróide v_i da seguinte forma

$$J_{FCM}(U, V; X) = \sum_{i=1}^{c} \sum_{k=1}^{n} u_{ik}^m d(x_k, v_i)^2. \tag{3.5}$$

Durante o processo de agrupamento, os centroides e os graus de pertinência são atualizados até que um critério de parada seja satisfeito. Este critério pode ser um número de iterações t_{max} e/ou um limite ε para a melhoria entre as partições *fuzzy* da iteração atual e anterior. Se este último critério for satisfeito, os centroides da iteração atual podem ser considerados como os pontos mais representativos de cada grupo. O algoritmo 1 descreve o FCM.

Algoritmo 1: O algoritmo FCM

Entrada: Dataset $X \subset \mathbb{R}^{nxp}$
Número de grupos c
Parâmetro $m > 1$
Limite $\varepsilon > 0$ e número máximo de iterações t_{max}

1 $t \leftarrow 0$;
Inicialize randomicamente $U^{(t)}$;
repita
2 $t \leftarrow t + 1$;
Calcule os centroides $V^{(t)}$ através da equação (3.4) ;
Atualize a matriz de partição *fuzzy* $U^{(t)}$ através da equação (3.3)
até $\|U^{t-1} - U^t\| \leq \varepsilon$ ou $t > t_{max}$;
Saída: $U^{(t)}, V^{(t)}$

Após a organização dos dataset utilizando o FCM, será necessário treinar o SMA com uma abordagem de aprendizagem por reforço. O agente SMA original espera por um único conjunto linear de *setplays*. Não é uma solução escalável para lidar com centenas de demonstrações, nem para lidar com a estrutura organizada em grupos. Portanto, é necessário fornecer uma política de seleção de *setplays* para ser utilizada pelo SMA. A próxima seção descreve os fundamentos da aprendizagem por reforço.

3.3 APRENDIZAGEM POR REFORÇO

A aprendizagem por reforço é uma abordagem de aprendizagem de máquina onde um ou mais agentes executam ações exploratórias em um ambiente, cujo modelo de funcionamento é desconhecido, e recebem uma recompensa para cada ação executada. Esta recompensa indica para o agente se a consequência daquela ação é boa ou ruim para que ele possa atingir seus objetivos. Assim, a recompensa pode ser positiva ou negativa (RUSSELL; NORVIG, 2021). A aprendizagem por reforço estuda como um agente autônomo, que tem percepção e ação no seu ambiente, pode aprender a escolher as melhores ações que devem ser executadas para atingir o seu objetivo (MITCHELL, 1997).

A figura 3.6 ilustra esquematicamente o funcionamento geral da aprendizagem por reforço. Num dado instante $\hat{t}$, um agente é capaz de observar, através dos seus sensores, o estado atual do ambiente $s_{\hat{t}}$. Neste momento, ele irá usar sua atual política de controle para selecionar uma possível ação $a_{\hat{t}}$ para executar neste instante. O agente executa a ação selecionada e irá novamente observar o ambiente atualizando o estado atual $s_{\hat{t}+1}$. O agente também recebe uma recompensa $r_{\hat{t}+1}$ que irá estimar a utilidade da ação executada em relação à expectativa do agente atingir seus objetivos.

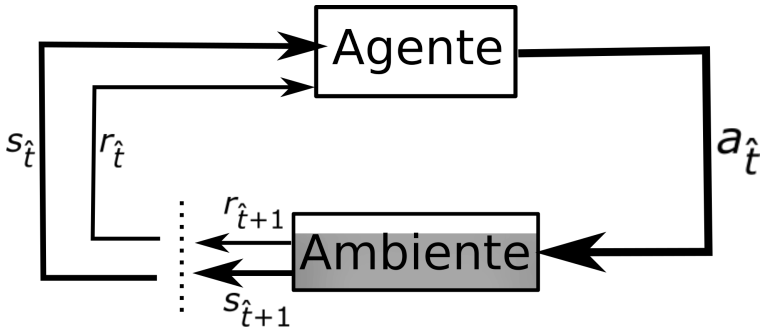


Figura 3.6: Modelo de Arquitetura utilizado por um agente que utiliza aprendizagem por reforço (SUTTON; BARTO, 2018). Em um instante $\hat{t}$, o agente observa o estado atual do ambiente $s_{\hat{t}}$ e seleciona uma ação $a_{\hat{t}}$. Após executar esta ação, observa o novo estado do ambiente $s_{\hat{t}+1}$ e recebe uma recompensa $r_{\hat{t}+1}$.

Nas próximas subseções são apresentados os fundamentos dos algoritmos que serão utilizados neste trabalho.

3.3.1 Aprendizagem Q

Um agente utilizando aprendizagem Q considera um problema modelado como (MITCHELL, 1997):

- O - conjunto de estados distintos que descrevem o seu ambiente, também conhecido como espaço de observação;
- A - conjunto de ações que o agente é capaz de executar, também conhecido com espaço de ação;
- $r(s_{\hat{t}}, a_{\hat{t}})$ - função de recompensa recebida pelo agente quanto executa uma ação $a_{\hat{t}}$ no instante $\hat{t}$ a partir do estado atual do ambiente $s_{\hat{t}}$, onde $a_{\hat{t}} \in A; s_{\hat{t}} \in O$;
- $\delta(s_{\hat{t}}, a_{\hat{t}})$ - função de transição do ambiente que determina qual o próximo estado do ambiente $s_{\hat{t}+1}$ quando a ação $a_{\hat{t}}$ for executada no estado atual $s_{\hat{t}}$.

O algoritmo aprendizagem Q é uma algoritmo livre de modelo, ou seja, ele não possui um modelo de como o ambiente se comporta. Isto significa que o agente desconhece as funções r e δ. A tarefa do agente é aprender uma política $\pi : O \longrightarrow A$ para selecionar a ação $a_{\hat{t}}$ para executar no estado observado pelo agente $s_{\hat{t}}$ no instante $\hat{t}$ (MITCHELL, 1997).

Entretanto, é necessário definir um critério para que o agente possa definir qual política π deve ser aprendida, ou seja, como avaliar a utilidade da política aprendida. Uma abordagem possível é induzir o agente a aprender a política π que gere a maior recompensa acumulada possível durante o tempo. Este requisito é representado pelo valor $V^{\pi}(s_{\hat{t}})$ definido como

$$V^{\pi}(s_{\hat{t}}) = \sum_{i=0}^{\infty} \gamma^{i} r_{\hat{t}+i1}, \tag{3.6}$$

onde a sequência de recompensas $r_{\hat{t}+i1}$ é obtida aplicando-se repetidamente a política π a partir do estado $s_{\hat{t}}$. $0 \leq \gamma < 1$ é uma constante que estipula o valor relativo entre as recompensas futuras e imediata. Recompensas recebidas i passos de tempo no futuro são descontadas exponencialmente por um fator de γ^{i}. Se $\gamma = 0$, apenas a recompensa imediata é considerada. Quanto mais γ aproxima-se de 1, maior ênfase atribui-se às recompensas futuras em relação à recompensa imediata. V^{π} é chamado recompensa cumulativa descontada (MITCHELL, 1997).

A tarefa do agente passa a ser aprender uma política π que maximize o valor de V^{π} para qualquer estado $s_{\hat{t}} \in O$. A política aprendida desta forma é chamada política ótima e é denotada por π^{*}:

$$\pi^{*} = \underset{\pi}{\operatorname{argmax}}\ V^{\pi}(s_{\hat{t}}), \forall s_{\hat{t}} \in O. \tag{3.7}$$

Define-se $V^{*}(s_{\hat{t}})$ como a máxima recompensa cumulativa descontada obtida ao seguir a política ótima iniciando no estado $s_{\hat{t}}$. É difícil para o agente aprender diretamente uma função $\pi^{*} : O \longrightarrow A$, uma vez que não possui dados de treinamento rotulados. O agente

possui apenas uma sequencia de recompensas obtidas à proporção que explora o ambiente. Neste contexto, é mais fácil aprender uma função numérica de avaliação sobre estados e ações do que aprender uma política a partir de uma função de avaliação (MITCHELL, 1997).

Define-se $Q(s, a)$ uma função de ação-utilidade diretamente relacionada com as funções de utilidade na forma

$$V(s) = \max_{a} \ Q(s, a). \tag{3.8}$$

$Q(s, a)$ denota o valor de executar a ação a no estado s. As funções Q agregam uma importante propriedade ao agente aprendiz: eles não precisam conhecer a função de transição do ambiente $\delta(s, a)$. Desta forma, a aprendizagem Q é uma estratégia livre de modelo (RUSSELL; NORVIG, 2021). O valor $Q(s, a)$ é uma estimativa do valor de recompensa total esperada iniciando-se no estado s e executando como primeira ação a.

A aprendizagem Q atualiza iterativamente os valores de Q, enquanto o agente explora o ambiente. Para isto utiliza a seguinte equação para atualização

$$Q(s_{\hat{t}}, a_{\hat{t}}) = (1 - \alpha)Q(s_{\hat{t}}, a_{\hat{t}}) + \alpha[r(s_{\hat{t}}, a_{\hat{t}}) + \gamma \max_{a_{\hat{t}+1}} \ Q(s_{\hat{t}+1}, a_{\hat{t}+1})]. \tag{3.9}$$

A atualização é realizada logo após o agente executar a ação $a_{\hat{t}}$ a partir do estado $s_{\hat{t}}$ e observar o novo estado $s_{t\hat{+}1}$, obtendo a recompensa $r(s_{\hat{t}}, a_{\hat{t}})$ (SUTTON; BARTO, 2018). $\alpha \in (0, 1]$ é a taxa de aprendizagem do agente que garante que $Q(s, a)$ seja atualizado considerando parte do valor atual e acrescente um novo valor em proporções complementares. É um requisito para a convergência da aprendizagem Q que α seja inicializado com valores maiores e vá decrescendo ao longo do tempo (SUTTON; BARTO, 2018). O algoritmo 2 descreve a aprendizagem Q.

Algoritmo 2: O algoritmo de aprendizagem Q (SUTTON; BARTO, 2018)

Entrada: $\alpha \in (0, 1]$
Inicialize $Q(s, a)$ arbitrariamente $\forall s \in O, a \in A$, exceto que $Q(terminal, .) = 0$;
$\hat{t} \leftarrow 0$;
para cada *episódio* **faça**
  Observe $s_{\hat{t}}$;
  repita
    Escolha $a_{\hat{t}}$ de A usando a política derivada de Q;
    Execute a ação $a_{\hat{t}}$;
    Observe $r(s_{\hat{t}}, a_{\hat{t}})$ e $s_{\hat{t}+1}$;
    Atualize $Q(s_{\hat{t}}, a_{\hat{t}})$ utilizando a equação (3.9);
    $s_{\hat{t}} \leftarrow s_{\hat{t}+1}$
  até $s_{\hat{t}}$ *é terminal*;
fim
Saída: Q

Uma questão que resta resolver é como representar a função Q. A forma mais direta é através de uma tabela. Entretanto, a tabela não é escalável para problemas com

dimensões, espaços de observação e de ação grandes, além de perder em generalidade. Diversas outras abordagens já foram apresentadas (RUSSELL; NORVIG, 2021)(SUTTON; BARTO, 2018), mas neste trabalho o enfoque é no uso de redes neurais profundas que serão o tema da próxima seção.

3.3.2 Aprendizagem Q profunda (Deep Q Network (DQN))

A função Q pode ser generalizada pela aproximação de funções lineares, mas também é comum o uso de funções não-lineares como as redes neurais. Denomina-se rede-**Q**, uma rede neural com pesos θ que aproxima a função Q. Uma rede-Q pode ser treinada ajustando os parâmetros $\theta_{\hat{t}}$ na iteração $\hat{t}$ para reduzir o erro quadrático médio na equação (3.9), onde os valores alvo ótimos $r(s_{\hat{t}}, a_{\hat{t}}) + \gamma \max_{a_{\hat{t}+1}} Q(s_{\hat{t}+1}, a_{\hat{t}+1})$ são substituídos por valores alvo aproximados $y_{\hat{t}} = r(s_{\hat{t}}, a_{\hat{t}}) + \gamma \max_{a_{\hat{t}+1}} Q(s_{\hat{t}+1}, a_{\hat{t}+1}; \bar{\theta}_{\hat{t}})$ com pesos $\bar{\theta}_{\hat{t}}$ de uma iteração anterior (MNIH et al., 2015).

O algoritmo DQN é definido aplicando-se o algoritmo tradicional de aprendizagem Q (ver seção 3.3.1) neste contexto, atualizando-se os pesos a cada intervalo de tempo, substituindo as recompensas esperadas por amostras simples e definindo $\bar{\theta}_{\hat{t}} = \theta_{\hat{t}-1}$. O DQN está descrito no algoritmo 3 (MNIH et al., 2015).

O DQN utiliza uma técnica chamada repetição de experiências (LIN, 1992), onde as experiências do agente são armazenadas a cada intervalo de tempo, $e_{\hat{t}} = (s_{\hat{t}}, a_{\hat{t}}, r_{\hat{t}}, s_{\hat{t}+1})$, em um conjunto de dados $D_t = \{e_1, \ldots, e_{\hat{t}}\}$, agrupados por vários episódios (onde o fim de um episódio acontece quando um estado terminal é alcançado) em uma memória de repetição. Durante o laço mais interno do algoritmo 3, aplica-se atualizações de aprendizagem Q, ou atualizações de minilotes, a amostras de experiências $e_i \in D_{\hat{t}}$ extraídas aleatoriamente da memória de repetição (MNIH et al., 2015).

O uso da repetição de experiências tem entre suas vantagens:

- cada experiência é potencialmente utilizada em várias atualizações de pesos da rede-Q, gerando maior eficiência de dados;
- aprender a partir de exemplos sequenciais é ineficiente, devido à forte correlação entre as amostras; extraindo amostras aleatórias, quebra-se esta correlação, reduzindo a variância das atualizações;

O algoritmo inicializa a memória de repetição até o seu limite superior N e as funções ação-valor Q e $\hat{Q}$ com pesos aleatórios θ. Para cada episódio, o estado inicial s_1 é observado através dos sensores do agente. Para cada passo de tempo $\hat{t}$, uma ação $a_{\hat{t}}$ é escolhida pelo agente de forma aleatória com probabilidade Ψ ou a ação como melhor valor de Q com probabilidade $1 - \Psi$. Quanto maior o valor de Ψ, mas o treinamento irá induzir o agente a explorar o espaço de ações, enquanto valores menores de Ψ levarão o agente a preferir usar a função ação-valor já aprendida até o momento.

Escolhida $a_{\hat{t}}$, o agente irá executá-la e observar a recompensa $r_{\hat{t}}$ e o novo estado $s_{\hat{t}+1}$. A transição $(s_{\hat{t}}, a_{\hat{t}}, r_{\hat{t}}, s_{\hat{t}+1})$ é armazenada na memória de repetição D. Em seguida, o algoritmo extrai uma amostragem de um minilote de transições (s_j, a_j, r_j, s_{j+1}) de D

Algoritmo 3: O algoritmo DQN com repetição de experiência (MNIH et al., 2015)

Inicialize a memória de repetição D até a capacidade N;
Inicialize a função ação-valor Q com pesos aleatórios θ;
Inicialize a função ação-valor alvo $\hat{Q}$ com pesos $\bar{\theta} = \theta$;
para cada *episódio* **faça**
  Observe o estado inicial s_1 ;
  para $\hat{t} \leftarrow 1$ **até** T **faça**
    Com probabilidade Ψ, selecione uma ação aleatória $a_{\hat{t}}$, caso contrário selecione $a_{\hat{t}} = \text{argmax}_a \, Q(s_{\hat{t}}, a; \theta)$;
    Execute a ação $a_{\hat{t}}$ e observe a recompensa $r_{\hat{t}}$ e o novo estado $s_{\hat{t}+1}$;
    Armazene a transição $(s_{\hat{t}}, a_{\hat{t}}, r_{\hat{t}}, s_{\hat{t}+1})$ em D;
    Tome uma amostragem de um minilote de transições (s_j, a_j, r_j, s_{j+1}) de D;

$$\text{Defina } y_j = \begin{cases} r_j & \text{se o episódio termina em } j+1 \\ r_j + \gamma \max_{a_{\hat{t}+1}} \hat{Q}(s_{j+1}, a_{\hat{t}+1}; \bar{\theta}) & \text{caso contrário} \end{cases}$$

    Execute um passo de gradiente descendente em $(y_j - Q(s_j, a_j; \theta))^2$ em relação aos parâmetros da rede θ;
    A cada C passos de tempo, reinicialize $\hat{Q} \leftarrow Q$
  fim
fim

e atualiza o valor alvo aproximado y_j, como descrito no algoritmo 3. Os pesos θ são atualizados por um passo de gradiente descendente em $(y_j - Q(s_j, a_j; \theta))^2$. A cada C passos de tempo a rede-$\hat{Q}$ é atualizada com os pesos da rede-Q.

O *DQN* é classificado como um algoritmo *off-policy* pois atualiza sua política a partir de experiências coletadas em políticas anteriores. Isto melhora a eficiência de amostragem durante o treinamento pois não descarta as experiências anteriores durante o treinamento (PRUDENCIO; MAXIMO; COLOMBINI, 2022). O DQN foi pioneiro na área de aprendizagem por reforço profunda ao ser utilizado para aprender a jogar diversos jogos do videogame Atari a partir dos pixels das imagens destes jogos (MNIH et al., 2015).

3.4 CONSIDERAÇÕES FINAIS

No próximo capítulo, descreve-se como o DQN foi utilizado para aprender qual o grupo de *setplays* é mais adequado para um time de futebol de robôs em cada situação de jogo que se apresenta. Os grupos foram gerados através de uma adaptação do FCM gerando um *dataset* organizado de *setplays*. Todo desenvolvimento de ferramentas necessárias pare coletar e gerar os setplays também está descrito no próximo capítulo.

Capítulo

4

Neste capítulo, descreve-se a metodologia de investigação adotada nesta tese. Todas as etapas, desde a construção do dataset até os experimentos e resultados obtidos, são apresentadas a seguir.

APRENDIZAGEM DE SETPLAYS POR DEMONSTRAÇÃO

O problema investigado nesta tese é o aprendizado de comportamentos cooperativos num Sistema Multiagentes (SMA) utilizando o conhecimento informal de assistentes humanos. Para solucionar este problema, foi desenvolvida uma abordagem avaliada sobre o ambiente *RoboCup 3D Soccer Simulation* (3DSSIM), reconhecido como referência na comunidade científica. Esta abordagem possui 3 etapas, conforme ilustrado na Figura 4.1, as quais são descritas a seguir.

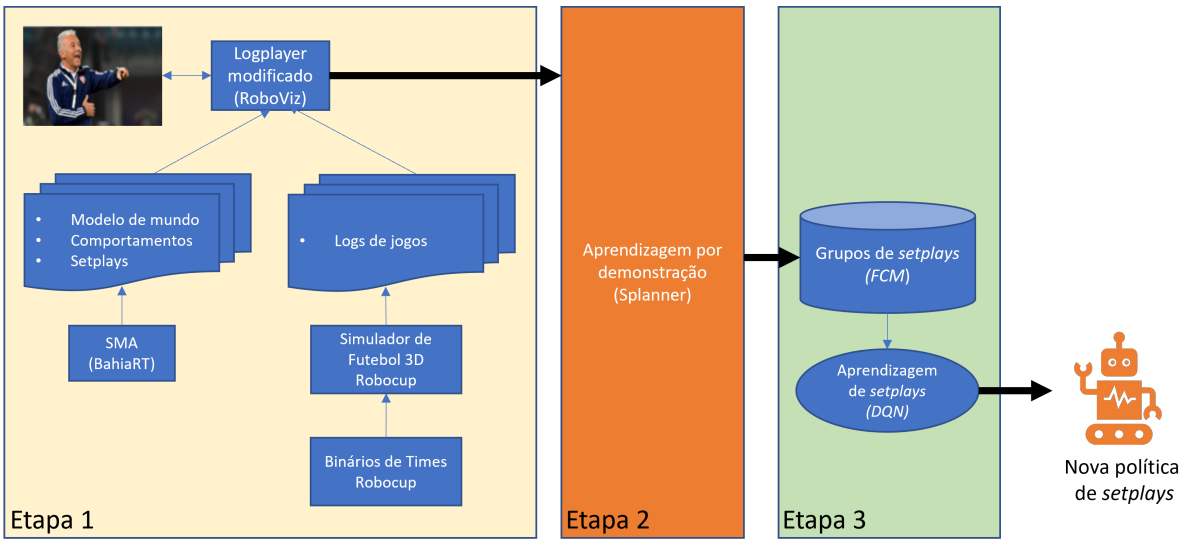


Figura 4.1: Aprendizagem de Setplays por Demonstração.

Na etapa 1, espera-se que especialistas ou não em futebol, assistam os jogos do SMA *Bahia Robotics Team* (BahiaRT)[1] em competições oficiais recentes a partir de uma modificação no *Logplayer* oficial da 3DSSIM (*RoboViz*)[2], fazendo pausas em situações que os mesmos considerem que os robôs simulados no BahiaRT tiveram um comportamento coletivo indesejado ou abaixo do desempenho que poderiam ter. As cenas das jogadas capturadas na etapa 1 são levadas para a etapa 2, onde ocorre a aprendizagem por demonstração (*Learning from Demonstration* (LfD) (ARGALL et al., 2009)). A Seção 4.1 descreve as modificações realizadas no *RoboViz* e seu uso para captura das cenas iniciais das demonstrações.

Na etapa 2, a ferramenta *Strategy Planner* (SPlanner) (CRAVO et al., 2014) foi alterada para que a mesma seja utilizada como um gerador de demonstrações de forma efetiva no ambiente 3DSSIM, tornando-a apta a iniciar um novo *setplay* (conforme definição apresentada na Seção 3.1.1) a partir da cena capturada no *RoboViz*. A Seção 4.2 descreve as adaptações realizadas no SPlanner e o novo modo de demonstração adicionado ao mesmo.

Para a realização das etapas 1 e 2, foi utilizada uma estratégia de *crowdsourcing* baseada num kit de ferramentas, denominado *BahiaRT Setplays Collecting Toolkit*, que integra as versões atualizadas do *RoboViz* e SPlanner com um formulário para envio de demonstrações pela comunidade e público em geral. A Seção 4.3 descreve a arquitetura e uso do kit de ferramentas.

Na etapa 3, os *setplays* coletados nas etapas anteriores compõem uma base de dados organizada por meio do agrupamento fuzzy com a finalidade de reunir em um mesmo grupo *setplays* semanticamente equivalentes ou similares. Para tal, foi realizada uma adaptação no algoritmo *Fuzzy C-Means* (FCM) que é apresentada na Seção 4.4.

Com a base de dados organizada, o algoritmo *Deep Q Network* (DQN) (conforme apresentado na seção 3.3.2) foi utilizado para treinar uma política de seleção do grupo de *setplays* mais eficaz para ser usado em cada situação de jogo. O modelo usado na aprendizagem por reforço está descrito na Seção 4.5.

4.1 ROBOVIZ: ADICIONANDO UM MODO DE DEMONSTRAÇÃO

O *RoboViz* é o visualizador oficial da competição 3DSSIM. O*RoboViz* é usado em dois modos operacionais: modos de jogo e log. O modo de jogo é usado como padrão nos jogos em tempo real durante as competições. O*RoboViz* se conecta a um simulador na rede e recebe todas as informações da cena de jogo, enquanto as mensagens enviadas pelos agentes são processadas pelo simulador. O*RoboViz* renderiza a cena gráfica correspondente ao instante atual do jogo e a exibe em um monitor.

No modo de log, não são necessários simuladores ou conexões online. O*RoboViz* abre um arquivo de log previamente gerado pelo simulador e renderiza as cenas correspondentes registradas neste arquivo de log. Do ponto de vista do usuário, é como uma repetição de uma partida disputada anteriormente.

O modo log é muito útil para desenvolvedores e pesquisadores que buscam entender

[1] https://www.acso.uneb.br/bahiart

[2] https://github.com/magmaOffenburg/RoboViz

as limitações de suas equipes para definir estratégias e melhorar seu desempenho. Eles podem assistir às partidas disputadas durante competições ou testes anteriores em seus laboratórios e detectar as limitações de suas equipes. No entanto, quando eles identificam algum ponto a ser aprimorado, eles precisam codificar manualmente as condições que representam aquela cena específica em suas ferramentas de desenvolvimento. Não há como extrair automaticamente essa situação do jogo para outras ferramentas.

Neste trabalho, um novo modo de demonstração foi adicionado ao *RoboViz* (SIMõES et al., 2021). Este modo pode ser usado ao iniciar o*RoboViz* usando o parâmetro –*demoMode*. Neste novo modo, o *RoboViz* é iniciado em uma interface semelhante ao modo de log, mas contendo um botão extra *Start a new Demonstration*, conforme mostrado na Figura 4.2. O botão está desativado quando o aplicativo é iniciado, mas quando um arquivo de log é aberto, o botão é habilitado. O *RoboViz* começa a reproduzir um log assim que o arquivo é aberto pelo usuário. Os botões restantes na janela do Logplayer são semelhantes aos de um player de vídeo comum, como reproduzir/pausar, retroceder, avançar e outros.

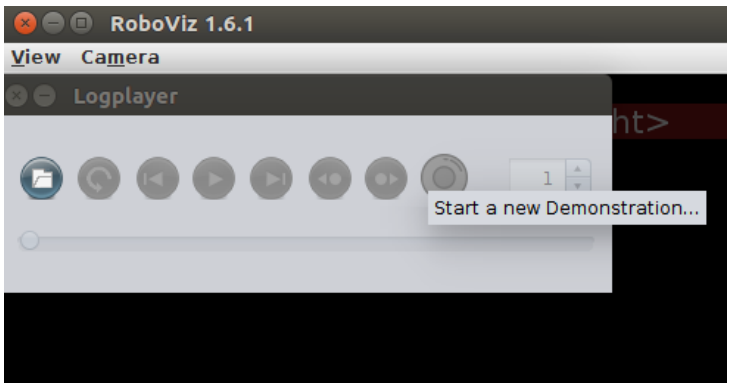


Figura 4.2: Tela inicial do*RoboViz* no modo de demonstração. O botão *Start a new Demonstration* pode ser usado durante a reprodução do log para criar uma nova demonstração.

Durante a reprodução do log, quando o usuário observa uma situação em que deseja iniciar uma nova demonstração, ele clica no botão e fornece algumas informações conforme mostrado na Figura 4.3. O usuário deve fornecer algumas informações iniciais nesta tela. Ele escolhe uma equipe (um SMA) para a qual está fazendo essa nova demonstração. Em seguida, ele escolhe o tipo de *setplay*, o qual pode ser ofensivo ou defensivo, e um modo de jogo (ver Tabela 4.1).

Tabela 4.1: Modos de jogo disponíveis em uma partida da 3DSSIM. Durante uma partida, *Left* e Right são substituídos pelo nome da equipe.

Modo de jogo	**Descrição**
Before KickOff	Descreve os momentos do jogo que antecedem um início ou reinício de partida. Por exemplo, início do primeiro ou segundo tempos ou quando um gol é marcado por uma das equipes.
KickOff Left/Right	São modos de jogo que indicam que a equipe da esquerda *Left* ou direita *Right* está autorizada a iniciar ou reiniciar a partida.
PlayOn	Descreve o modo de jogo padrão, ou seja, jogo em andamento normal.
KickIn Left/Right	Cobrança de lateral favorável à equipe do lado esquerdo/direito. Na 3DSSIM a cobrança de lateral é feita com os pés.
CornerKick Left/Right	Cobrança de escanteio favorável à equipe do lado esquerdo/-direito.
GoalKick Left/Right	Cobrança de tiro de meta favorável à equipe do lado esquerdo/direito.
FreeKick Left/Right	Cobrança indireta de falta favorável à equipe do lado esquerdo/direito. A Cobrança indireta significa que um gol só pode ser marcado depois que dois jogadores da equipe que cobra a falta tocarem na bola. Não é possível chutar direto ao gol. Na 3DSSIM não há cobrança direta de falta (onde é possível chutar direto ao gol).
Pass Left/Right	Modo de passe ativado favorável à equipe do lado esquerdo/-direito. Este modo é ativado quando um agente de um dos times envia um comando *Pass* para o simulador. Se satisfeitas condições específicas (ajustadas a cada revisão anual de regras[3], o simulador ativa este modo impedindo que qualquer oponente aproxime-se num raio de um metro da bola. É um modo exclusivo da 3DSSIM para estimular o desenvolvimento das habilidades de troca de passes pelas equipes.
Goal Left/Right	Indica que um gol foi marcado pela equipe do lado esquerdo/-direito.
GameOver	Indica o final da partida.

Quando o tipo ofensivo é selecionado, a ferramenta assume que o time escolhido possui posse de bola. Caso contrário, considera que a equipe adversária é a dona da bola. A lista de modos de jogo é sensível ao contexto. Apenas os modos de jogo que foram observados no log atual até o instante em que a reprodução foi interrompida, serão mostrados. Se, por exemplo, não houve tiro de meta na partida atual até o momento, a

[3]https://ssim.robocup.org/3d-simulation/3d-rules/)

lista atual não mostrará o tiro de meta como opção. O usuário deve escolher o modo de jogo correspondente ao ponto para o qual deseja que o log seja reposicionado. Se você escolher *Kick-in BahiaRT*, por exemplo, o log reverterá para o último *kick-in* (cobrança de lateral) favorável ao BahiaRT.

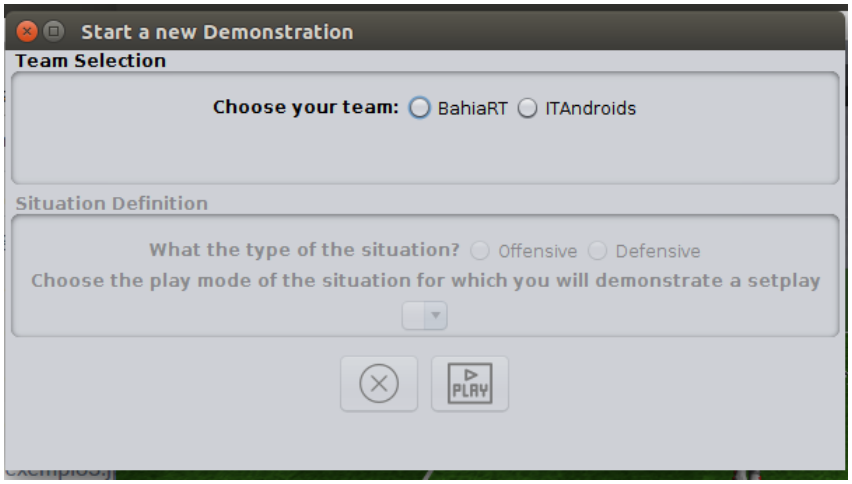


Figura 4.3: Tela iniciada para começar uma nova demonstração.

Quando o usuário preencher todas as informações, ele pode pressionar o botão *Play* para reposicionar o log. Este processo pode ser mais lento do que o que se espera de um reprodutor de vídeos, porque a entrada para o *RoboViz* não é um *streaming* de vídeo. O *RoboViz* lê um log textual gerado pelo simulador e busca reproduzir as cenas da simulação ocorrida. Assim, sempre que um reposicionamento do log for necessário, deve-se construir novamente a sequência de cenas necessária para atingir o ponto alvo no log. Uma nova janela é exibida com instruções para aguardar o log reverter para a posição de destino e pressionar o botão *Pause* quando o log estiver na cena em que o usuário deseja iniciar a demonstração. Os botões *forward/backward* podem ser usados para ajustar a posição exata no log. A Figura 4.4 mostra a janela de definição do instante de início da demonstração.

Quando o log é reposicionado, o usuário pode escolher os jogadores do time que ele escolheu na tela anterior que farão parte do *setplay*. Chamamos esses jogadores de companheiros de equipe. O usuário apenas clica em cada jogador e o número de cada robô selecionado é adicionado a uma lista ao lado da opção *Teammates selected*. Quando todos os jogadores estiverem selecionados, o usuário deve marcar a caixa à esquerda desta opção. Em seguida, o usuário deve selecionar os jogadores da equipe adversária que deseja incluir no *setplay* e marcar a opção *Opponnets selected*.

O usuário pressiona o botão *Start demonstration* e um arquivo de demonstração é exportado para o disco. O SPlanner é iniciado automaticamente para importar este arquivo. A Figura 4.5(a) apresenta um exemplo de cena de demonstração no *RoboViz*, pouco antes da exportação. Os números azuis e vermelhos na janela no canto superior direito indicam os jogadores escolhidos para participar da demonstração. Já na Figura

Figura 4.4: Após o reposicionamento do log, os companheiros de equipe são escolhidos. O usuário apenas clica em cada jogador e seu número é inserido em uma lista ao lado da opção *Teammates selected*. A caixa de seleção deve ser marcada quando todos os companheiros de equipe forem escolhidos e o processo se repete para os oponentes.

4.5(b) tem-se um exemplo de cena de demonstração no SPlanner logo após a importação. Observe que os jogadores apresentados no *RoboViz* são carregados na mesma posição relativa no campo no SPlanner.

Quando o SPlanner importar o arquivo de demonstração, os jogadores considerados companheiros de equipe serão representados por camisetas brancas, e os oponentes usarão camisetas azuis. A partir deste ponto, o usuário projetará seu *setplay* usando recursos regulares do SPlanner. Ao terminar, ele exporta os *setplays* para um arquivo e fecha o SPlanner. O*RoboViz* continuará a reprodução do log no mesmo ponto em que a demonstração foi iniciada. Assim, um usuário pode gerar várias demonstrações, independentes umas das outras, enquanto assiste à uma única partida.

O arquivo exportado pelo *RoboViz* usa uma sintaxe de expressão-S possível de ser analisada por outras ferramentas além do SPlanner. Um exemplo do arquivo é apresentado na Figura 4.6.

O arquivo contém as seguintes informações: duas listas de jogadores - companheiros de equipe (*players*) e oponentes (*playerOpponents*); modo de jogo (*condition*), o jogador líder (*leadPlayer*) e o jogador com posse da bola (*ballHolder*). O modo de jogo é o escolhido pelo usuário ao iniciar a demonstração. O jogador líder é sempre um companheiro de equipe. Em jogadas ofensivas, o líder e o detentor da posse de bola são o mesmo jogador. Nas jogadas defensivas, o líder é o companheiro de equipe que está mais próximo da bola, enquanto o detentor da posse da bola é o oponente que está mais próximo da bola.

Esta seção apresentou um dos produtos desta tese publicado em (SIMõES et al., 2021). O resultado consiste em uma nova versão para o *RoboViz*, no qual foi adicionado um novo modo de demonstração que permite aos usuários capturarem cenas de jogos assistidos para usar em outras ferramentas para planejamento estratégico. Após a aprovação desta tese,

Figura 4.5: (a) Demonstração no *RoboViz* pronta para ser exportada. (b) Demonstração importada para o SPlanner.

a versão final do RoboViz será atualizada para compatibilidade com a última versão no repositório oficial da ferramenta e será publicado neste mesmo repositório, representando uma contribuição pública direta para a comunidade científica.

Na próxima seção, os ajustes e melhorias realizados na ferramenta *SPlanner* para viabilizar este trabalho são apresentados.

4.2 SPLANNER: GERANDO DEMONSTRAÇÕES DE SETPLAYS

O foco principal desta seção é descrever como o SPlanner foi adaptado para tornar-se útil para preencher conjuntos de dados(SIMõES; SILVA; NOGUEIRA, 2020) com base em demonstrações de especialistas no domínio do futebol robótico. Estas modificações também tornaram o SPlanner mais adequado para uso na competição 3DSSIM e preencheram algumas lacunas que limitavam seu uso em partidas de futebol robótico.

O SPlanner foi originalmente desenvolvido pela equipe FCPortugal (CRAVO et al., 2014) e atualizado posteriormente (MOTA et al., 2015) juntamente com o *FCPortugal Setplays Framework* (FSF) disponível através da *libSetplays* (FABRO; REIS; LAU, 2014). Parte desta solução não possui os requisitos necessários para o desenvolvimento amplo de *setplays* defensivos e ofensivos, considerando requisitos comuns de jogo em futebol de robôs humanoides. Sendo assim, como um dos resultados desta tese, o SPlanner e o FSF foram modificados (SIMõES et al., 2021)(SIMõES et al., 2020) para suprir as tais lacunas.

As melhorias adicionadas ao SPlanner foram desenvolvidas conforme apresentado nas subseções a seguir.

```
(demo :name BahiaRT-400.525
      :players
            (list
                  (at (playerRole :roleName Player7) (pt :x 6.890516 :y -1.2782754))
                  (at (playerRole :roleName Player8) (pt :x 4.9517903 :y -1.0762119))
                  (at (playerRole :roleName Player9) (pt :x 6.243064 :y 0.3161399))
                  (at (playerRole :roleName Player11) (pt :x 7.497106 :y 0.45839706))
            )
      :playersOpponents
            (list
                  (at (playerRole :roleName PlayerOpponent1) (pt :x 14.6746235 :y -0.2478271))
                  (at (playerRole :roleName PlayerOpponent2) (pt :x 9.516699 :y -0.60560226))
                  (at (playerRole :roleName PlayerOpponent3) (pt :x 8.434658 :y -1.3939612))
                  (at (playerRole :roleName PlayerOpponent4) (pt :x 10.192728 :y -1.9074758))
                  (at (playerRole :roleName PlayerOpponent5) (pt :x 9.348128 :y 1.4145325))
                  (at (playerRole :roleName PlayerOpponent6) (pt :x 5.5126295 :y -0.67984545))
                  (at (playerRole :roleName PlayerOpponent11) (pt :x 6.4658666 :y -1.640686))
            )
      :condition (playm PlayOn)
      :leadPlayer (playerRole :roleName Player7)
      :ballHolder (playerRole :roleName Player7)
)
```

Figura 4.6: O arquivo de demonstração exportado por *RoboViz* usa uma sintaxe de expressão-S.

4.2.1 Equipe Adversária

Algumas ações defensivas dependem de um jogador adversário. Por exemplo, comportamentos de marcação ou retomada de bola, precisam ter jogadores oponentes como referência. Assim, a possibilidade de colocar adversários nos *setplays* foi adicionada para viabilizar essas ações, como pode ser visto na Figura 4.7.

Figura 4.7: Banco de jogadores adversários.

Uma equipe completa de 11 adversários usando camisas azuis foi adicionada junto com uma bancada para eles na lateral do campo. Cada oponente tem três comportamentos (ações) possíveis, conforme menu apresentado na Figura 4.8.

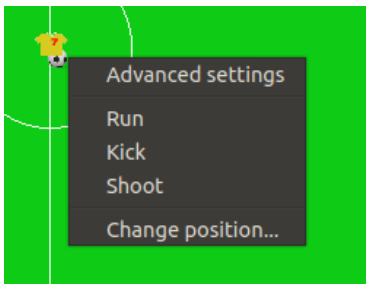


Figura 4.8: Menu de comportamentos dos jogadores oponentes.

- *Run:* move para uma posição alvo. Esta ação oferece ao designer de *setplays* a opção de estimar os movimentos dos oponentes de uma etapa de *setplays* para a seguinte. Esta ação pode ser usada tanto em jogadas ofensivas quanto defensivas.

- *Kick:* chuta a bola para outro jogador adversário. O objetivo é deixar o usuário estimar os passes feitos pela equipe adversária. Este comportamento está disponível apenas para o oponente que possui a bola em jogadas defensivas.
- *Shoot:* o designer de *setplays* pode usar esse comportamento para estimar uma situação em que um oponente pode chutar para o gol. Este comportamento está disponível apenas para o adversário que possui a bola em jogadas defensivas.

4.2.2 Jogadas Defensivas

Havia uma opção de *setplay* defensivo na versão anterior do SPlanner (CRAVO et al., 2014), mas a versão disponível desta ferramenta não a implementa. Os *setplays* defensivos são tão cruciais para a estratégia da equipe quanto os*setplays* ofensivos. Portanto, o SPlanner foi alterado para implementar efetivamente os *setplays* defensivos em todas as situações de jogo em que um *setplay* ofensivo estava disponível. Essa opção ampliou muito a capacidade da equipe de futebol robótico de realizar jogadas coletivas a qualquer momento em uma partida. Uma característica importante para dar suporte aos *setplays* defensivos é a presença de jogadores adversários, uma vez que os comportamentos defensivos podem usar os adversários como referência. Por exemplo, um companheiro de equipe pode realizar uma ação de marcação defensiva durante uma bola parada para seguir um adversário em campo e impedi-lo de receber um passe.

4.2.3 Novos Comportamentos

O SPlanner tinha oito ações disponíveis que poderiam ser executadas por cada jogador em sua versão base, como pode ser visto na Figura 4.9 (CRAVO et al., 2014).

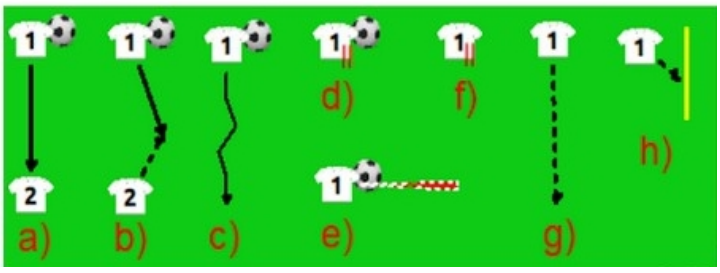


Figura 4.9: Ações disponíveis na versão base do SPlanner: a) Pass, b) Pass forward to <número do jogador>, c) Dribble, d) Hold, e) Shoot, f) Wait, g) Run, and h) Go to offside line.

- *Pass:* realizar um passe para um companheiro específico.
- *Pass forward to <número do jogador>:* passar para um ponto avançado no campo para onde o companheiro escolhido deve se mover interceptar a bola, recebendo o passe.
- *Dribble:* conduzir a bola evitando os adversários.

- *Hold:* parar na posição atual, protegendo a bola e mantendo a sua posse.
- *Shoot:* chutar em direção ao gol adversário.
- *Wait:* ficar parado no mesmo lugar.
- *Run:* mover-se para uma posição especificada.
- *Go to offside line:* mover-se para uma posição avançada, logo atrás da linha.

Para tornar o SPlanner mais adequado para à 3DSSIM, bem como preencher o conjunto de dados necessário para o mecanismo de aprendizagem por demonstração, novos comportamentos foram adicionados. A Figura 4.10 (a) mostra o menu de comportamentos modificados para jogadas ofensivas para jogadores sem posse de bola. A Figura 4.10 (b) mostra o menu de comportamentos modificados para jogadas ofensivas para jogadores com posse de bola.

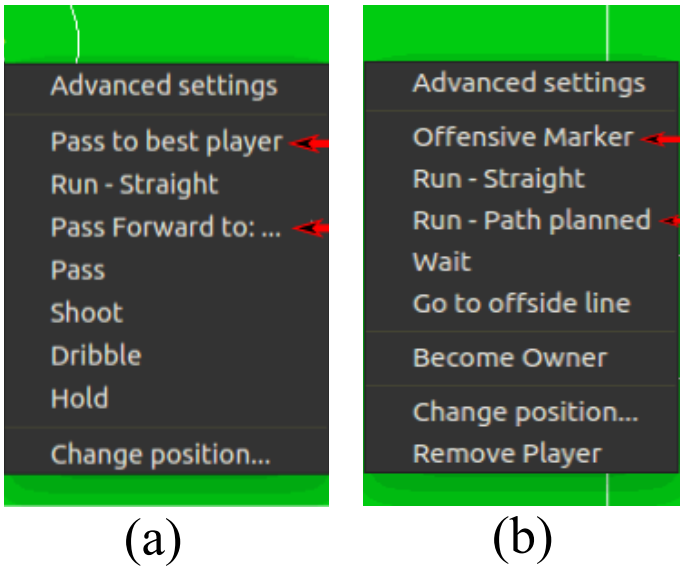


Figura 4.10: Novos comportamentos para jogadas ofensivas: (a) menu de comportamentos para jogadores sem posse de bola. (b) menu de comportamentos para jogadores com posse de bola.

Cada um dos novos comportamentos adicionados ao SPlanner são destacados por setas vermelhas na Figura 4.10 e descritos a seguir.

- *Run - Straight*: esta ação é um novo método de caminhada/corrida. Muitas equipes têm pelo menos dois tipos de caminhada: uma é mais lenta e confiável, com pouca ou nenhuma queda, e a outra é mais rápida, mais próxima de uma habilidade de corrida, mas com menos equilíbrio. *Run - Straight* é um comportamento criado para mapear o comportamento de caminhada/corrida mais rápido presente na equipe. Esse comportamento, na maioria dos casos, prioriza a velocidade ao invés

do equilíbrio e, geralmente, o jogador não está preocupado em prevenir colisões. É um movimento útil no planejamento estratégico quando o designer percebe que há um corredor de espaço livre onde um jogador pode correr em sua maior velocidade para surpreender adversários.

- *Run - Path planned*: o nome deste comportamento foi modificado de *Run* para *Run - Path Planned* para que o usuário possa ter clara a diferença entre esta ação de andar/correr para o comportamento *Run - Straight*. No *Run - Path Planned*, o jogador prioriza o equilíbrio em relação à velocidade e busca evitar colisões, ou seja, tem um movimento mais cuidadoso do que no *Run - Straight*. Assim, o designer terá duas opções de movimento de andar/correr para escolher quando utilizar cada um de forma estratégica.

- *Offensive Marker*: esta ação foi retirada do conjunto de comportamentos do BahiaRT que serviu de base para os estudos iniciais desta tese (SIMõES; NOGUEIRA, 2018). O objetivo principal deste comportamento é marcar agentes inimigos que impõem o risco de tomar a posse de bola de um dos nossos companheiros. O jogador se move para uma posição estratégica entre o inimigo e o companheiro com a bola, bloqueando o adversário e mantendo uma certa distância do jogador com a bola para que não afete a liberdade do mesmo para chutar, fazer um passe ou conduzir a bola.

- *Pass to best player*: Assim como o *Offensive Marker*, esta ação também vem do BahiaRT. Este comportamento delega para o time usar um algoritmo de seleção para escolher o aliado em melhores condições para receber um passe. Quando usada corretamente, esta ação pode resultar em excelentes jogadas em um jogo. Quando o designer usa esse comportamento, ele deve criar várias transições para a etapa atual em que esse comportamento é usado. Cada transição leva o *setplay* para o próximo passo, considerando um companheiro de equipe diferente escolhido para receber o passe. Este tipo de *setplay* é chamado multi-fluxo.

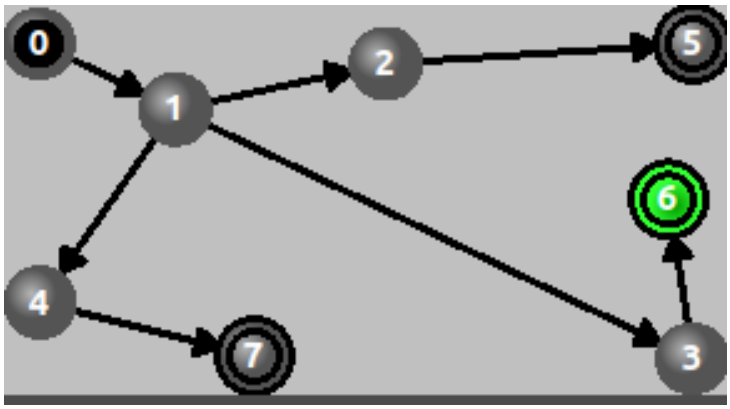


Figura 4.11: Máquina de Estados representando um setplay multi-fluxo com uso do comportamento Pass to best player.

A Figura 4.11 ilustra um exemplo de setplay multi-fluxo utilizando o comportamento *Pass to best player*. O estado 1 possui três transições possíveis partindo dele. Cada transição corresponde a um jogador selecionado pelo agente passador para receber a bola. A Figura 4.12 ilustra cada uma das três transições neste exemplo. Na Figura 4.12a o projetista definiu a transição para o caso em que o agente com a bola selecionar o jogador mais distante na sua lateral esquerda. Neste caso toda a sequencia da jogada se dará considerando esta decisão do agente com a bola no estado 1. Da mesma forma, as figuras 4.12b e 4.12c ilustram as sistuações em que o agente escolhe o recebedor como um jogador próximo à esquerda e o jogador mais distante à direita, respectivamente. O *Pass to best player* permite a construção de setplays mais avaçados com múltiplos fluxos como este ilustrado no exemplo das Figuras 4.11 e 4.12.

(a) Primeira transição do estado 1 com o comportamento Pass to best player.

(b) Segunda transição do estado 1 com o comportamento Pass to best player.

(c) Terceira transição do estado 1 com o comportamento Pass to best player.

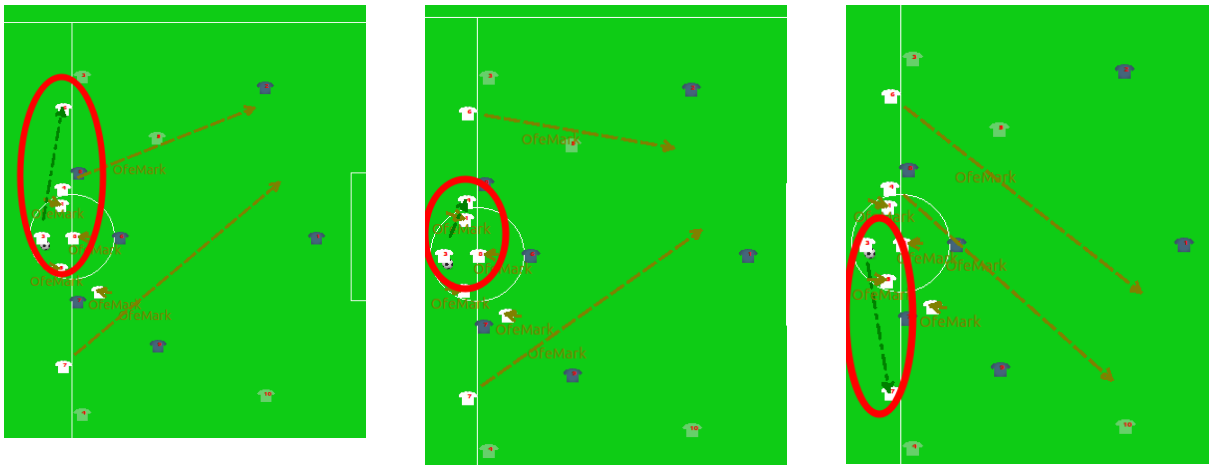


Figura 4.12: Exemplo de Setplay com comportamento Pass to best player.

- *Intercept*: esse comportamento não é exibido em nenhum menu de comportamentos. É um comportamento implícito relacionado à ação *Pass Forward To <número do jogador>*. Quando o usuário atribui o *Pass Forward To <número do jogador>* e seleciona um companheiro de equipe para receber a bola, este recebedor é implicitamente atribuído à ação *Intercept*. O receptor deve se mover para a região onde o jogador que possui a bola passa a bola e depois tenta interceptá-la. Nesta versão, o *Intercept* recebe como parâmetro a região para onde a bola é passada. Essa modificação torna mais fácil para as equipes implementarem esse comportamento ao estender o FSF.

Alguns novos comportamentos também foram projetados para situações defensivas e

podem ser observadas na Figura 4.13, destacados por setas vermelhas. Essas novas ações são descritas a seguir. O comportamento *Run - Straight*, descrito anteriormente, também está disponível no menu dos *setplays* defensivos.

- *Defensive Marker*: esta ação está relacionada com a marcação de adversários em situações defensivas. No SPlanner ela aparece com uma única ação e pode ser aplicada em relação a oponentes com ou sem a posse de bola. Na implementação do BahiaRT estendendo o FSF, este comportamento é dividido em dois tipos: o marcador ativo quando o jogador marca o adversário com a posse de bola tentando recuperar a posse de bola; e o marcador passivo quando o jogador marca um potencial receptor de um passe executado pelo adversário. Desta forma, este comportamento se propõe a impedir qualquer jogada do adversário. Quando uma equipe estende o FSF, ela decide por fornecer as duas especializações (ativa e passiva) do comportamento ou um único comportamento para os dois casos.

- *Become Owner*: esta ação muda o dono da bola para o jogador que executa este comportamento. Ela foi projetada para uso em jogadas defensivas. O objetivo deste comportamento é fazer com que um companheiro de equipe tome a posse de bola de um adversário. Em geral, esta é a última ação utilizada em uma jogada defensiva, uma vez que o objetivo principal de uma estratégia defensiva é recuperar a posse de bola para o time que executa a jogada.

Figura 4.13: Novos comportamentos para jogadas defensivas.

4.2.4 Condições de cancelamento de Passo

Na versão base do SPlanner, este campo foi inicialmente chamado de “Step Times” (CRAVO et al., 2014). Mas, percebeu-se a necessidade de adicionar um campo

"Step Abort Conditions". Como o "Step Times"também era um campo de cancelamento do passo, decidiu-se reutilizar o campo alterando o nome e acrescentando o que era necessário para generalizar as condições de cancelamento do passo. Em um *setplay*, existem condições para que ele comece a executar (condições de início), condições que asseguram sua continuidade (condições de transição) e condições que uma vez atendidas devem cancelar a execução do *setplay*(condições de cancelamento). O SPlanner tinha condições gerais de cancelamento, por exemplo, quando o inimigo rouba a bola. Porém, não haviam condições específicas de cancelamento para cada passo do *setplay*, e essas eram necessárias, pois existem situações durante uma etapa específica do jogo em que as condições desejáveis não são atendidas, devendo o *setplay* ser encerrado. Isso ocorre, por exemplo, por uma tentativa falhada de passe. Para corrigir isso, a opção "passFailed"foi adicionada ao campo "Step Abort Conditions", que encerra o setplay em caso de passe falho na etapa atual do *setplay*, conforme apresentado na Figura 4.14).

Figura 4.14: Novo campo "Step Abort Conditions"com a condição "passFailed".

Todas essas mudanças no SPlanner devem ser espelhadas no FSF pois as mudanças na *libSetplay*[4] permitem que os agentes de um SMA estendam os comportamentos presentes no SPlanner e forneçam sua implementação para cada comportamento. Por exemplo, a maneira como o BahiaRT implementa o *Run - Straight* pode ser diferente do algoritmo usado por outra equipe.

Quando o desenho de um *setplay* é finalizado no SPlanner, o usuário pode exportá-lo para um arquivo de texto usando uma linguagem específica definida pelo FSF (CRAVO et al., 2014). Essa linguagem é baseada em uma sintaxe de expressões-S contendo um *setplay* para ser interpretado pelo FSF e executado por um Sistema Multiagente, como por exemplo, o BahiaRT. A Figura 4.15 mostra as relações entre SPlanner, FSF e BahiaRT.

Figura 4.15: SPlanner gera um arquivo de expressões-S contendo um *setplay* para ser interpretado pelo FSF e executado pelo BahiaRT.

[4] *libSetplay* é uma biblioteca de código aberto onde o FSF é implementado. Está disponível em https://bitbucket.org/bahiart3d/libsetplay

A linguagem usada para representar o *setplay* em um arquivo foi estendida para suportar os novos comportamentos descritos nesta seção. A sintaxe para essas ações novas e modificadas é exibida na Figura 4.16.

```
defMark:(defMark : destPlayer <PLAYER-REFERENCE>: relTo <TYPE> )

passtobestplayer:(pbplayer :player <PLAYER-REFERENCE> :type <TYPE>)

run Straight: (mov :region <REGION>)

intercept: (intercept :region <REGION>)

ofensive marker: (ofeMark : destPlayer <PLAYER-REFERENCE>: relTo <TYPE> )
```

Figura 4.16: Sintaxe de expressões-S das novas ações a serem interpretadas pelo FSF.

Esta sintaxe apresenta a forma como as ações são geradas e lidas pelo FSF. Esta extensão na *libSetplay* torna todas as melhorias apresentadas neste trabalho disponíveis para qualquer equipe robótica que possa estender o FSF.

Após as modificações no SPlanner e no FSF, em acordo com os autores originais, os dois ambientes passaram a ser mantidos publicamente no repositório do grupo de pesquisa do autor desta tese. A reorganização destas duas ferramentas como software livre e disponível à comunidade é mais um resultado deste trabalho.

Com as modificações no FSF, os agentes do SMA BahiaRT foram atualizados para que pudessem executar os novos comportamentos acrescentados ao SPlanner. O ferramental descrito nesta seção e na seção anterior é utilizado neste trabalho na estratégia de *crowdsourcing* para construção do dataset de *setplays* usado para aprendizagem por demonstração. Na próxima seção, é apresentado o conjunto de ferramentas e a estratégia de coleta dos *setplays* para construção do dataset.

4.3 BAHIART SETPLAYS COLLECTING TOOLKIT

A construção da base de dados com *setplays* é baseada numa estratégia de *crowdsourcing* em que pessoas de qualquer parte do mundo podem contribuir fazendo papel do especialista humano.

Para dar suporte a esta estratégia, construiu-se um conjunto de ferramentas reunindo as soluções apresentadas nas seções 4.1 e 4.2. Para tornar fácil a instalação e utilização das ferramentas, as mesmas foram organizadas em *containers docker*[5], evitando que os usuário precisem compilar as ferramentas, instalar bibliotecas e resolver problemas com dependências. Este conjunto organizado de ferramentas para coleta de *setplays* foi denominado **BahiaRT Setplays Collecting Toolkit** e disponibilizado publicamente junto com toda a documentação necessária para publicação no repositório ⟨https://bitbucket.org/bahiart3d/setplaysdataset⟩.

Os requisitos para instalar o BahiaRT Setplays Collecting Toolkit são:

[5]https://www.docker.com/

- Instalar o git e o docker em uma distribuição Linux moderna.
- Para usuários de placas gráficas NVIDIA, atualizar os drivers NVIDIA. Verificar se o sistema operacional está usando os drivers NVIDIA corretamente.
- Para usuários sem placa gráfica NVIDIA, atualizar os drivers da placa gráfica.
- ***apenas para usuários do Windows*** Instalar e configurar o Docker Desktop, WSL2 e um X Server para Windows. (Disponível apenas para sistemas Windows 10 ou +).

Atendidos os requisitos, a instalação é realizada seguindo os comandos:

1. *git clone https://bitbucket.org/bahiart3d/setplaysdataset.git*
2. *cd setplaysdataset*
3. *./setup.sh nvidia*, se tiver uma placa gráfica NVIDIA; ou *./setup.sh mesa* para outras configurações

Este procedimento funciona em qualquer distribuição Linux baseada em Debian. Para outras distribuições, pode ser necessário, antes de executar o passo 3, editar o arquivo *setup.sh* para substituir o gerenciador de pacotes pelo que é usado na distribuição em questão (ex: trocar o *apt* por yum ou *zypper*).

Para usuários Windows 10+, recomenda-se usar uma máquina virtual WSL2 com sistema operacional Linux Ubuntu 18.04 ou superior. Nesta máquina virtual, será possível executar o procedimento acima para instalação do BahiaRT Setplays Collecting Toolkit. Neste caso, é preciso iniciar o X Server antes de seguir os próximos passos para a execução.

O BahiaRT Setplays Collecting Toolkit é um container docker composto dos seguintes itens:

- RoboViz com o modo de demonstração ativo;
- SPlanner com o modo de demonstração ativo e os novos comportamentos criados;
- Repositório com 19 jogos disputados em competições oficiais nos anos de 2021 e 2022;
- Repositório *setplays* vazia para serem armazenadas as demonstrações criadas.

Uma vez instalado o *container* apropriado, a execução é realizada com os comandos *run.sh nvidia* ou *run.sh mesa*, de acordo com a instalação que tenha sido realizada.

Ao executar o *container*, automaticamente será exibida a tela inicial do RoboViz em modo de demonstração (ver Figura 4.2). A partir daí, os procedimentos já explicados na seção 4.1 devem ser seguidos para capturar uma situação inicial de jogo para uma demonstração. Para encontrar jogos para assistir, o usuário deve buscar o repositório *games* dentro do *container* que irá encontrar os 19 jogos distribuídos junto com o BahiaRT Setplays Collecting Toolkit. Usuários com experiência no uso do *docker* podem copiar

outros jogos para dentro deste repositório no *container* caso queiram usar outros logs de jogos como base para as demonstrações.

Ao iniciar a demonstração, o SPlanner será aberto numa tela inicial para escolha das zonas do campo às quais a demonstração é aplicável, conforme ilustra a figura 4.17.

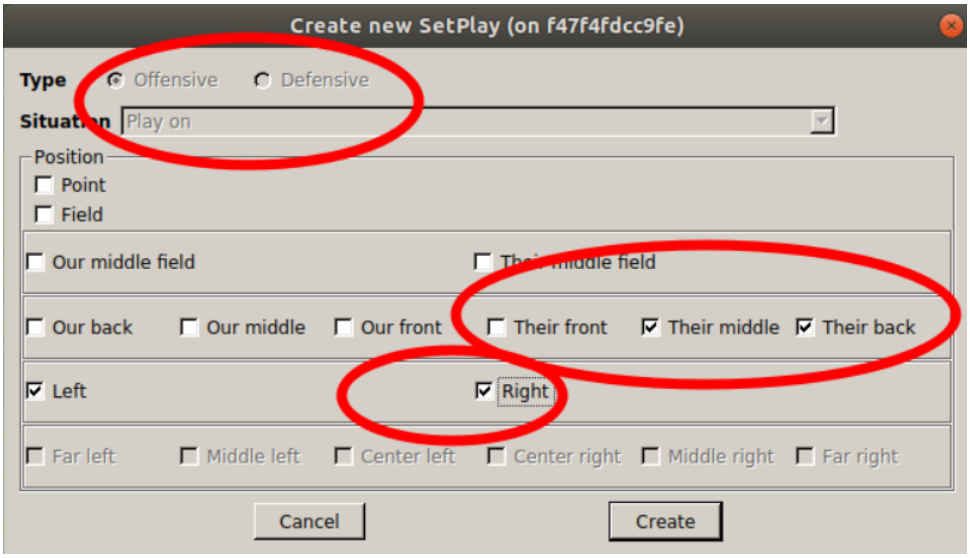


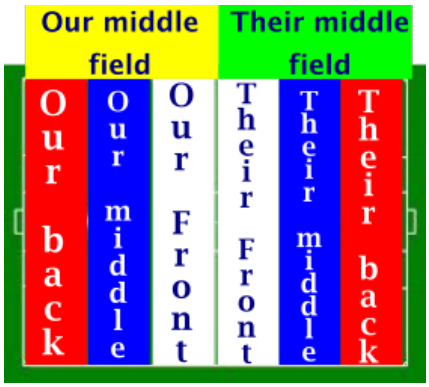
Figura 4.17: Tela inicial do SPlanner após carregar uma cena inicial do RoboViz para iniciar uma nova demonstração.

As opções de tipo (*Type*) e situação (*Situation*) da nova demonstração já vem pre-enchidas com as escolhas feitas no RoboViz e não podem ser alteradas. Nesta tela, o usuário deve escolher as regiões do campo às quais esta demonstração pode se aplicar. O SPlanner considera uma divisão de campo nos eixos horizontal e vertical, sendo a primeira parte na tela divisão horizontal e a segunda parte (inferior) a vertical. As zonas são geradas pela interseção entre as zonas escolhidas no eixo horizontal e vertical. As opções das zonas estão descritas através da Figura 4.18

Na Figura 4.18a são exibidas as regiões no eixo horizontal, iniciando na defesa em direção ao ataque são: *Our back, Our middle, Our front, Their front, Their middle, Their back*. As três regiões prefixadas por *Our* também podem ser referenciadas conjuntamente por uma região única chamada *Our middle field* que representa todo o campo defensivo. Da mesma forma o *Their middle field* representa todo o campo ofensivo com suas três sub-regiões.

A Figura 4.18b exibe as regiões do eixo vertical, da equerda para a direita são: *Far left, Middle left, Center left, Center right, Middle right, Far right*. As três primeiras regiões podem ser agrupadas numa grande região chamada *Left* e as três últimas são reunidas na região *Right*. Para considerar a referência de esquerda e direita, considere um jogador de pé no meio do campo posicionado de frente para o gol adversário. Nesta orientação, é que se considera o que se chama de esquerda e direita nestas regiões do eixo vertical.

(a) Regiões do Eixo Horizontal



(b) Regiões do Eixo Vertical

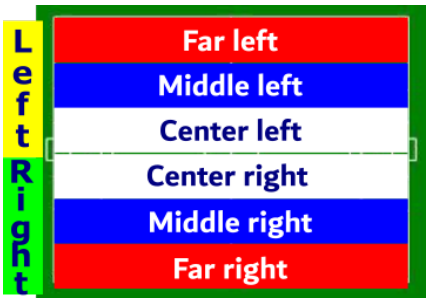


Figura 4.18: Divisão do campo em regiões no SPlanner.

A escolha das regiões na tela da Figura 4.17 será o resultado da interseção entre as regiões escolhidas no eixo horizontal com aquelas escolhidas no eixo vertical. Por exemplo, na figura 4.17 foi selecionado *Their middle*, *Their back* e *Right*, então a demonstração poderá ser ativada em qualquer lugar da intermediária ofensiva pelo lado direito do campo.

Depois desta tela inicial, o primeiro passo do *setplay* que será demonstrado será exibido na tela (ver Figura 4.19). A partir daí, o demonstrador só precisa utilizar os menus com os comportamentos descritos na seção 4.2 para ir construindo sua demonstração.

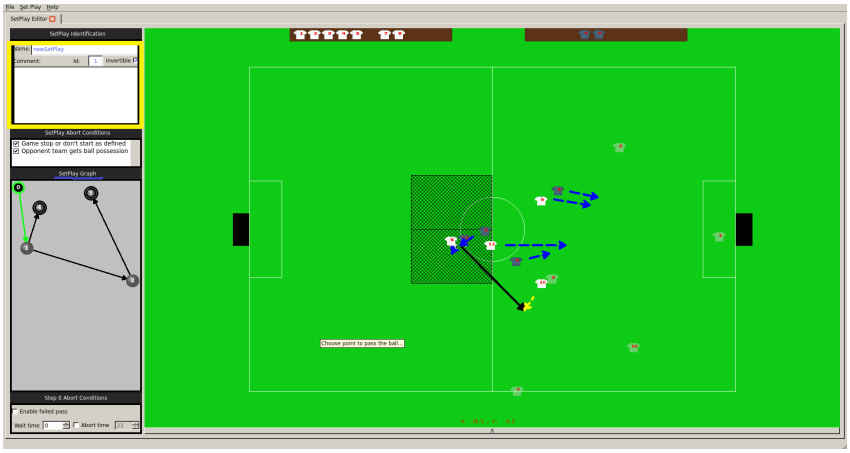


Figura 4.19: Tela inicial com o primeiro passo do *setplay* que será demonstrado.

O BahiaRT Setplays Collecting Toolkit possui documentação detalhada em texto[6] e video[7] explicando como usar a ferramenta para gerar demonstrações.

Para cada demonstração concluída, deve-se gerar um arquivo contendo a descrição na sintaxe de expressões-S para o *setplay* definido, por meio do menu *File - Export setplay*.

[6] ⟨https://bitbucket.org/bahiart3d/setplaysdataset/src/master/USAGE.md⟩
[7] ⟨https://youtu.be/h_s8rA2IS88⟩

Será exibida uma tela como a da Figura 4.20 e o usuário deve localizar uma pasta de nome *setplays* para salvar sua demonstração. O salvamento nesta pasta é fundamental, pois esta é a única pasta do *container* que está mapeada para a maquina hospedeira. Salvando as demonstrações nesta pasta, será possível recuperá-las depois que a aplicação for encerrada. Caso contrário, todo o trabalho será perdido.

Depois de salva a demonstração, o SPlanner pode ser encerrado. Isto não encerra o BahiaRT Setplays Collecting Toolkit. O controle retornará para o RoboViz no mesmo instante da partida que estava sendo assistida, quando foi iniciada a demonstração. O usuário pode continuar assistindo a partida e gerando novas demonstrações, conforme o procedimento descrito nesta seção. Ao final de cada partida, um novo log pode ser aberto e novas partidas podem ser assistidas. Cada demonstrador pode gerar quantas recomendações de *setplays* desejar, assistindo uma ou várias partidas.

Quando o demonstrador não desejar mais assistir jogos, pode encerrar o RoboViz. Isto irá fechar o container do BahiaRT Setplays Collecting Toolkit. O demonstrador encontrará suas recomendações na pasta *./setplays/* a partir da pasta de trabalho que estava quando executou o script *run.sh*. Para enviar suas contribuições, basta compactar tudo num arquivo *.zip* e submeter pelo formulário de submissão que acompanha essas ferramentas: ⟨https://formfaca.de/sm/fB2AVqE8m⟩

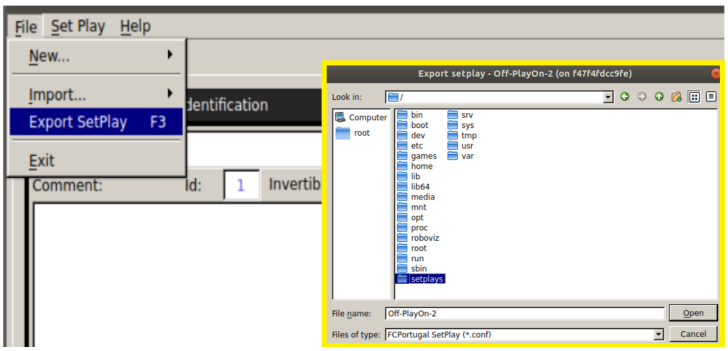


Figura 4.20: As demonstrações devem ser salvas na pasta *setplays* para que possam ser recuperadas depois que a aplicação for encerrada.

Após a aprovação desta tese, todo o dataset com os *setplays* enviados será tornado público e continuará sendo atualizado periodicamente sempre que novas demonstrações forem enviadas.

Durante quatro meses foram coletadas 382 demonstrações com diferentes origens. Não é exigida a identificação dos colaboradores de forma que algumas demonstrações são anônimas e outras têm seus autores identificados. Para a estratégia *crowdsourcing* a identificação do autor dos *setplays* não é relevante, sendo o mais importante a diversidade para buscar aumentar as chances de generalizar políticas de *setplays* mais eficazes de acordo com as habilidades da equipe que irá usá-los.

A grande quantidade de *setplays* inviabiliza o uso direto pelo FSF que não possui gerenciadores de *setplays* escaláveis dentro das restrições temporais do simulador da 3DSSIM. Na próxima seção, é descrita a solução para organização deste dataset,

dividindo-o em grupos por similaridade, para viabilizar a aprendizagem de uma política de seleção de *setplays* adequada ao volume apresentado.

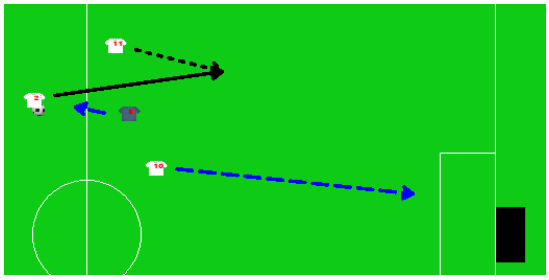
4.4 ORGANIZANDO O DATASET DE SETPLAYS

A estratégia adotada para construir o dataset para este trabalho provê uma expectativa de trabalhar com um número indefinidamente alto de instâncias para compor o dataset de *setplays*. Entretanto, com muitas pessoas diferentes gerando demonstrações a partir de jogos de futebol de robôs, as chances de ter *setplays* equivalentes é alta. Não seriam *setplays* exatamente iguais, pois, muitos dos atributos que compõem um *setplay* possuem valores reais num espaço contínuo, reduzindo a probabilidade da igualdade absoluta. Então, define-se que esta igualdade entre os *setplays* irá existir quando houver uma **equivalência semântica**.

Definição 1 (**Equivalência Semântica**). *Dois setplays*SP_i *e* $SP_j, i \neq j$, *são considerados semanticamente equivalentes se representarem a mesma jogada no nível de conhecimento abstrato do domínio.*

Para melhor compreender o que significa a mesma jogada no nível do conhecimento abstrato do domínio, considere os dois *setplays* SP_i e SP_j na Figura 4.21. Os dois *setplays* representam uma jogada bem conhecida no futebol chamada triangulação. Na triangulação, um jogador faz um passe para um companheiro que está mais a frente e ao lado para desviar a bola de um oponente que se aproxima. Enquanto isto, um terceiro companheiro avança para receber um novo passe mais adiante deixando-o em condição de chutar a gol.

(a) SP_i: triangulação inciando pelo companheiro à esquerda.



(b) SP_j: triangulação iniciando pelo companheiro à direita.

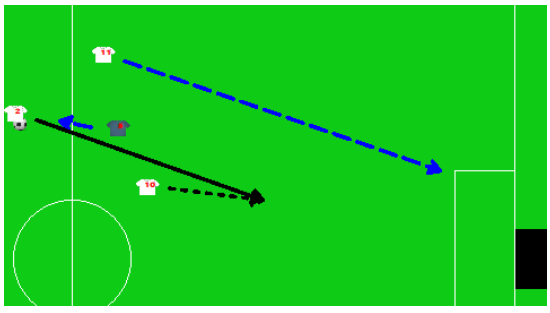


Figura 4.21: Duas jogadas de triangulação semanticamente equivalentes que têm por objetivo ultrapassar o oponente levando a bola para uma posição próxima à área adversária.

As duas situações exibidas na figura 4.21 são idênticas, entretanto as jogadas não são exatamente iguais. Na Figura 4.21a, a jogada SP_i inicia com um passe para o companheiro que está adiantado no lado esquerdo para uma futura conclusão da triangulação com o avanço do companheiro que está no lado direito. Já na Figura 4.21b, a jogada SP_j inicia

com um passe para o companheiro que está avançado pelo lado direito com uma futura conclusão da triangulação com o avanço do companheiro que está no lado esquerdo. Apesar das diferenças, no nível abstrato do domínio do futebol são duas jogadas de triangulação capazes de superar o oponente e gerar uma oportunidade de gol, ou seja, são jogadas semanticamente equivalentes.

Apresentado o conceito, estima-se que num dataset de *setplays* coletados com a estratégia descrita na seção 4.3, uma grande quantidade de *setplays* semanticamente equivalentes estejam presentes. Soma-se a isto o fato de que a escolha de *setplays* num conjunto grande para uma aplicação com restrições temporais como o futebol de robôs, torna-se inviável. Para solucionar esta questão, decidiu-se por dividir o dataset, buscando organizá-lo em grupos divididos com base na similaridade dos *setplays*.

Como o próprio conceito de equivalência semântica requer uma interpretação num nível de abstração do domínio do conhecimento, é bastante previsível que um determinado *setplay* tenha características que o assemelhem a mais de um grupo ao mesmo tempo. Portanto, optou-se pelo uso do FCM como uma solução para organização do dataset em grupos *fuzzy* (SIMõES; SILVA; NOGUEIRA, 2020).

Um esquema de conjunto de dados foi definido para agrupar os *setplays* extraídos da análise dos arquivos com sintaxe de expressão-S coletados com a estratégia definida na seção 4.3. Dos campos contidos no arquivo, foram consideradas as propriedades listadas na Tabela 4.2 para representar os *setplays*.

Tabela 4.2: Propriedades do dataset extraídas dos *setplays* gerados como demonstração através do BahiaRT Setplays Collecting Toolkit.

Propriedade	**Descrição**	**Tipo de dado**
ourPlayersNumber	Número dos jogadores do BahiaRT participando do *setplay*	Inteiro
theirPlayersNumber	Número de jogadores oponentes participando do *setplay*	Inteiro
abortCondition	Expressão booleana representando a condição de cancelamento do *setplay*	Árvore binária representando a expressão booleana equivalente
Steps	Número total de passos que compõem o *setplay*	Inteiro
stepsList	Lista de passos que compõem o *setplay*	Vetor de passos. Cada passo é formado pela estrutura representada na Tabela 4.3.

A última propriedade (*stepsList*) na Tabela 4.2 é uma lista de estruturas complexas compostas por vários propriedades. Um conjunto de dados secundário é descrito na Tabela 4.3, detalhando cada passo que integra o atributo *stepsList*.

Tabela 4.3: Propriedades que compõem um passo de um *setplay*.

Propriedade	**Descrição**	**Tipo de dado**
ourPlayersInStep	Número de jogadores do BahiaRT participando neste passo. $ourPlayersInStep \leq ourPlayersNumber$	Inteiro
theirPlayersInStep	Número de jogadores oponentes participando deste passo. $theirPlayersInStep \leq theirPlayersNumber$	Inteiro
waitTime	Tempo mínimo a esperar antes que uma transição do passo atual para um seguinte possa ser conduzida	Real
abortTime	Tempo máximo para terminar o passo atual.	Real
ourPlayersList	Lista de jogadores do BahiaRT participando deste passo.	Vetor de jogadores P_B. $\forall p_B \in P_B; p_b \sim (x_{p_B}, y_{p_B})$, onde (x_{p_B}, y_{p_B}) é um par de coordenadas cartesianas que representa a posição aproximada de p_b no início deste passo.
theirPlayersList	Lista de jogadores oponentes participando deste passo.	Vetor de jogadores P_O. $\forall p_O \in P_O; p_o \sim (x_{p_O}, y_{p_O})$, onde (x_{p_O}, y_{p_O}) é um par de coordenadas cartesianas que representa a posição aproximada de p_o no início deste passo.
nextStep	Identificação do próximo passo após uma condição de transição ser satisfeita.	Inteiro
condition	Expressão booleana que determina a condição de transição para o próximo passo	Árvore binária representando a condição booleana equivalente.
behaviorsList	Lista de comportamentos executadas por cada jogador $p_B \in P_B$ enquanto o *setplay* estiver neste passo.	Vetor de strings.

A Figura 4.22 ilustra uma visão esquemática do modelo de conjunto de dados proposto. O esquema é dividido em dois níveis. No primeiro nível, existem características

que identificam os *setplays*. Cada instância neste nível representa um *setplay* diferente recomendado pelos demonstradores. O segundo nível descreve os passos dentro de cada *setplay*. Na fase de extração, os arquivos de *setplays* são lidos e transformados na estrutura ilustrada na Figura 4.22.

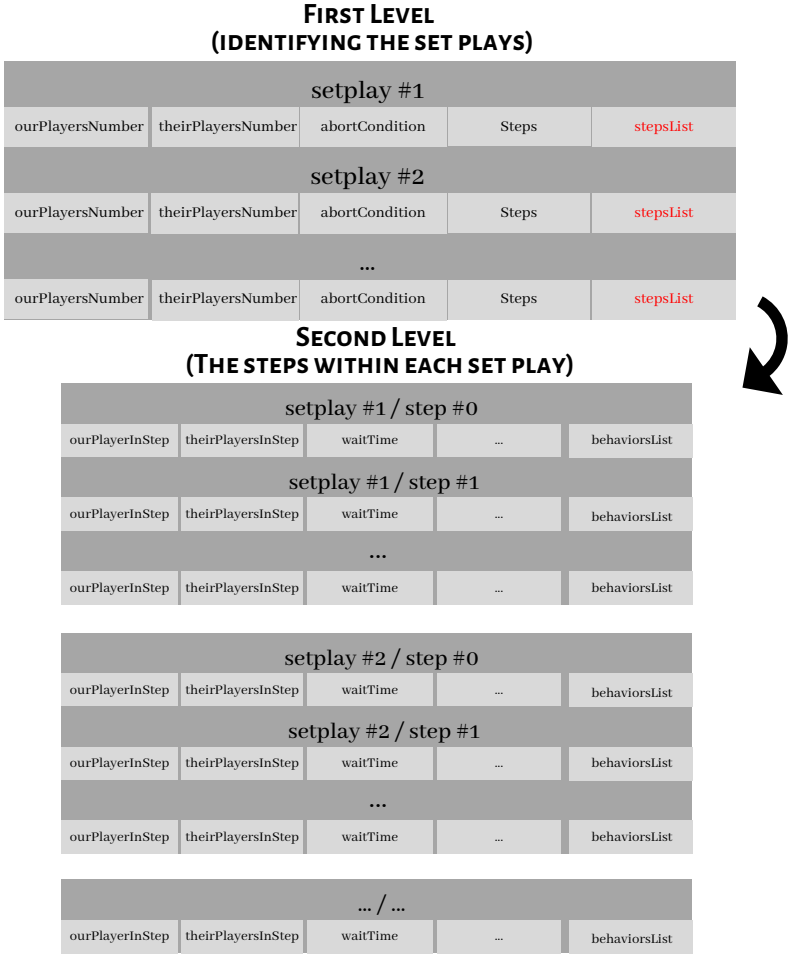


Figura 4.22: Representação esquemática do dataset. O primeiro nível contém as propriedades da Tabela 4.2 que identificam os *setplays*. O segundo nível contém as propriedades da Tabela 4.3 que descrevem os passos de cada setplay.

Para realizar o agrupamento do conjunto de dados organizado neste esquema, o FCM foi utilizado em duas etapas.

Na primeira etapa, são consideradas apenas as propriedades do primeiro nível do esquema do conjunto de dados, exceto o atributo *stepsList*. O primeiro estágio gera um novo conjunto de dados é nomeado *Dataset Agrupado 1*. Este dataset contém um conjunto de clusters de *setplays* semanticamente equivalentes, considerando apenas a similaridade entre os quatro primeiros atributos utilizados no primeiro nível do esquema do dataset. A ideia é que *setplays* com um número semelhante de passos, condições de cancelamento e números jogadores participantes possam ser agrupados no mesmo grupo. Como está

sendo utilizada uma abordagem Fuzzy, um *setplay* pode fazer parte de mais de um grupo com diferentes graus de pertinência.

Na segunda etapa, várias instâncias do FCM foram executadas, uma para cada cluster do *Dataset Agrupado 1*. O objetivo é considerar cada cluster como o conjunto de dados base para cada execução de FCM. Como resultado, o *Dataset Agrupado 1* contém clusters que serão divididos em subclusters. O conjunto de todos os subclusters gerados neste segundo estágio forma o *Dataset Agrupado 2*. Nesta segunda etapa, o algoritmo realiza uma avaliação refinada da equivalência semântica. Agora, os recursos no segundo nível do esquema do conjunto de dados são considerados. *Setplay* com descrições de passos semelhantes podem ser agrupados no mesmo subgrupo.

A motivação para dividir a solução em duas etapas foi reduzir a complexidade de avaliar a equivalência semântica. Como o próprio conceito de equivalência semântica é ambíguo, é fornecido um viés para a solução de agrupamento. Inicialmente, apenas as características do primeiro nível do esquema foram utilizadas para fomentar que *setplays* com diferenças importantes no número de jogadores ou passos não sejam agrupados no mesmo cluster de equivalência semântica. Além disso, *setplays* com diferenças consideráveis na condição de cancelamento não são agrupados no mesmo cluster. Vale ressaltar que, como trata-se de uma abordagem fuzzy, é possível obter alguns *setplays* com diferenças nessas características em relação às demais instâncias no mesmo cluster, mas apresentando um menor grau de pertinência.

Para utilizar o FCM, é preciso definir uma medida de distância apropriada para medir a similaridade entre as instâncias do dataset. A distância euclidiana é comumente usada para estimar a distância entre duas instâncias com propriedades escalares. Entretanto, o esquema de dataset proposto contém alguns tipos de dados não escalares. Define-se a seguir uma medida apropriada para o problema apresentado nesta tese.

Considere um conjunto de instâncias $X = \{x_1, x_2, \ldots, x_n\}$, sendo n o número de instâncias no conjunto de dados X. Cada instância $x_i = \{x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, x_{i,5}\}$, $\forall i = 1, \ldots, n$ é um vetor que representa as propriedades descritas na Tabela 4.2. Logo, $x_{i,1}$ equivale à propriedade *ourPlayersNumber*, $x_{i,2}$ equivale à propriedade *theirPlayersNumber*, $x_{i,3}$ equivale à propriedade *abortCondition*, $x_{i,4}$ equivale à propriedade *Steps*, e $x_{i,5}$ equivale à propriedade *stepsList*.

Sendo assim, a distância entre duas instâncias x_i e $x_j, \forall i, j = 1, \ldots n; i \neq j$, é definida por:

$$d(x_i, x_j) = d(x_j, x_i) = \sqrt{(x_{j,1} - x_{i,1})^2 + (x_{j,2} - x_{i,2})^2 + \|x_{j,3} - x_{i,3}\|^2 + (x_{j,4} - x_{i,4})^2} \tag{4.1}$$

Observe que as propriedades $x_{i,5}$ e $x_{j,5}$ não são consideradas no primeiro estágio da solução FCM. Por esta razão, seus valores não são usados para calcular a distância entre x_i e x_j.

A propriedade *abortCondition*, representada por $x_{i,3}$ e $x_{j,3}$ na Equação 4.1, precisam de uma norma especial porque são árvores binárias que representam expressões booleanas. Todas as expressões booleanas neste conjunto de dados são geradas automaticamente do SPlanner a partir de um conjunto finito de condições. A estrutura das árvores binárias

geradas apresenta algumas variações. Portanto, a norma para $x_{i,3}$ e $x_{j,3}$ pode ser definida como:

$$\|x_{j,3} - x_{i,3}\| = DiffNode\,(x_{i,3}, x_{j,3}) \tag{4.2}$$

onde $DiffNode\,(x_{i,3}, x_{j,3})$ é o número de nós diferentes nas árvores representadas em $x_{i,3}$ e $x_{j,3}$. As demais propriedades são escalares e a norma é calculada de forma semelhante à distância euclidiana.

As normas definidas nas Equações (4.1) e (4.2) são utilizadas para calcular a função objetivo e atualização iterativa da matriz de partição do algoritmo FCM.

Na segunda etapa da execução do FCM, serão consideradas as instâncias da primeira etapa já agrupadas, *Dataset Agrupado 1*, $GD^{(1)}$. $GD^{(1)} = \{\psi_1, \ldots, \psi_{C^{(1)}}\}$, onde $C^{(1)}$ é o número de clusters gerados na primeira etapa. Cada cluster gerado na primeira etapa, $\psi_i, i = 1, \ldots, C^{(1)}$ será o conjunto de dados utilizado para a execução de cada instância do FCM na segunda etapa. O conjunto de dados da segunda etapa, portanto, é definido por $\psi_i = Y_{i,k} = \{y_{i,k,1}, y_{i,k,2}, \ldots, y_{i,k,\ddot{m}_k}\}$; onde $\ddot{m}_k$ é o número de passos do k-ésimo *setplay* no cluster ψ_i e $k = 1, \ldots, G_i$; onde G_i é o número total de *setplays* pertencentes ao conjunto de dados ψ_i.

Dados dois *setplays* no ψ_i, $Y_{i,k}$ e $Y_{i,l}$, a distância entre $Y_{i,k}$ e $Y_{i,l}$ é definida como:

$$d\,(Y_{i,k}, Y_{i,l})^{(2)} = d\,(Y_{i,l}, Y_{i,k})^{(2)} = d\,(Y_{i,l}, Y_{i,k})^{(1)} + \sum_{z=1}^{\min(\ddot{m}_k, \ddot{m}_l)} \|y_{i,l,z} - y_{i,k,z}\| \tag{4.3}$$

onde $d\,(Y_{i,l}, Y_{i,k})^{(1)}$ e $d\,(Y_{i,l}, Y_{i,k})^{(2)}$ são as distâncias entre os *setplays* $Y_{i,l}$ e $Y_{i,k}$ no primeiro e segundo estágio de agrupamento, respectivamente, no conjunto de dados ψ_i. $d\,(Y_{i,l}, Y_{i,k})^{(1)}$ é calculado conforme definido em (4.1). $z = 1, \ldots, \min(\ddot{m}_k, \ddot{m}_l)$ é o número do passo dos *setplays* $Y_{i,l}$ e $Y_{i,k}$.

Cada passo em ambos os *setplays* é comparado, computando a distância entre eles. Cada passo $y_{i,k,z}$ ou $y_{i,l,z}$ é um vetor de propriedades descritas na Tabela 4.3. $y_{i,k,z} = \{p_{i,k,z,1}, \ldots, p_{i,k,z,9}\}$ e $y_{i,l,z} = \{p_{i,l,z,1}, \ldots, p_{i,l,z,9}\}$ representam os 9 atributos que descrevem cada passo. A distância entre dois passos é definida por:

$$\|y_{i,l,z} - y_{i,k,z}\| = \sqrt{(p_{i,l,z,7} - p_{i,k,z,7})^2 + \sum_{j=5,6,8,9} \|p_{i,l,z,j} - p_{i,k,z,j}\|^2 + \sum_{j=1}^{4} (p_{i,l,z,j} - p_{i,k,z,j})^2} \tag{4.4}$$

Seguindo os mesmos critérios usados na primeira etapa, as propriedades com tipos de dados escalares (ex. $p_{i,l,z,j}$ e $p_{i,k,z,j}; j = 1, 2, 3, 4, 7$) utilizam um cálculo semelhante ao utilizado na distância euclidiana. As demais propriedades utilizam normas específicas de acordo com seu tipo de dado. $p_{i,l,z,8}$ e $p_{i,k,z,8}$ são árvores binárias com expressões booleanas. Sua norma é definida usando o mesmo método usado em (4.2).

As propriedades $p_{i,l,z,9}$ e $p_{i,k,z,9}$ são vetores de strings que representam os comportamentos de cada jogador nos passos $y_{i,k,z}$ e $y_{i,l,z}$. A norma compara os dois vetores e conta o número de comportamentos diferentes. A norma para $p_{i,l,z,9}$ e $p_{i,k,z,9}$ é definida como:

$$\|p_{i,l,z,9} - p_{i,k,z,9}\| = \sqrt{\sum_{j=1}^{\max(\bar{B}_{i,l,z}, \bar{B}_{i,k,z})} \Delta_j};$$
$$\Delta_j = \begin{cases} 1, if\ \beta_{i,l,z,j} \neq \beta_{i,k,z,j} \\ 0, otherwise \end{cases} \tag{4.5}$$

onde $\bar{B}_{i,l,z}$ e $\bar{B}_{i,k,z}$ são o número de comportamentos nos vetores $p_{i,l,z,9}$ e $p_{i,k,z,9}$, respectivamente; Δ_j é um acumulador; $\beta_{i,l,z,j}$ e $\beta_{i,k,z,j}$ são strings individuais que representam comportamentos nos vetores $p_{i,l,z,9}$ e $p_{i,k,z,9}$.

As demais características $p_{i,l,z,5}$, $p_{i,k,z,5}$, $p_{i,l,z,6}$ e $p_{i,k,z,6}$ são listas de jogadores. Cada jogador é definido por uma coordenada cartesiana pelo par ordenado (x, y), onde x e y são as regiões onde cada jogador deve estar no início de cada etapa. A distância euclidiana pode definir a distância entre dois jogadores. A soma de todas as distâncias entre pares de jogadores define a norma dessas listas de jogadores.

O algoritmo FCM deve ser executado tanto no primeiro quanto no segundo estágio usando as normas definidas para gerar um conjunto de clusters e subclusters representando o conceito de equivalência semântica.

Um parâmetro importante do processo de agrupamento é a quantidade de clusters. Para escolha da quantidade adequada de clusters que garantam o bom agrupamento de uma base de dados, é comum a utilização de um Índice de Validação de Clusters (IVC) sobre os agrupamentos obtidos com diferentes quantidades de clusters.

Muitos Índices de Validação de Clusters (IVCs) foram propostos e analisados em trabalhos recentes (EUSTáQUIO; NOGUEIRA, 2018)(EUSTáQUIO et al., 2018). Esses IVCs são usados para avaliar a qualidade da organização dos dados em clusters. Para esta tese foi escolhida a utilização do índice Silhueta Fuzzy (SF) (CAMPELLO; HRUSCHKA, 2006), uma vez que o mesmo tem viés não monotônico, boa escalabilidade para grandes conjuntos de dados e baixos custos de computação.

Para definir o conceito da SF, considere a matriz de partição fuzzy $U = [u_{i,j}]_{cxn}$; onde c é o número de clusters usados no algoritmo FCM, n é o número de objetos a serem organizados em clusters, $u_{i,j}$ é o grau de pertinência do objeto j ao grupo i, $i = 1, \ldots, c$, $j = 1, \ldots, n$. A SF é definida por (CAMPELLO; HRUSCHKA, 2006):

$$SF = \frac{\sum_{j=1}^{n} (u_{p,j} - u_{q,j})^{\lambda} \sigma_j}{\sum_{j=1}^{n} (u_{p,j} - u_{q,j})^{\lambda}}, \tag{4.6}$$

onde $u_{p,j}$ e $u_{q,j}$ são o primeiro e o segundo maiores elementos da j-ésima coluna da matriz de partição fuzzy, respectivamente, e λ é um coeficiente de ponderação definido pelo usuário. σ_j é a silhueta do objeto j definido da seguinte forma:

$$\sigma_j = \frac{\bar{b}_{p,j} - \bar{a}_{p,j}}{\max\left(\bar{a}_{p,j}, \bar{b}_{p,j}\right)}, \tag{4.7}$$

onde $\bar{a}_{p,j}$ é a distância média do objeto j para todos os outros objetos *pertencentes* ao cluster p. A distância é calculada usando as normas definidas anteriormente. Considera-se o j-ésimo objeto pertencente ao cluster p quando o grau de pertinência do j-ésimo objeto ao p-ésimo cluster fuzzy, $u_{p,j}$, é maior do que o grau de pertinência desse objeto a qualquer outro cluster fuzzy, ou seja, $u_{p,j} > u_{q,j} \forall q \in \{1, \ldots, c\}, q \neq p$. Seja $d_{q,j}$ a distância média do objeto j para todos os objetos *pertencentes* a outros clusters q, $q \neq p$. $\bar{b}_{p,j}$ é o $d_{q,j}$ mínimo calculado sobre $q = 1, \ldots, c$, $q \neq p$. O expoente λ é um parâmetro opcional definido pelo usuário ($\lambda = 1$, por padrão). Quando λ se aproxima de zero, a medida SF definida em (4.6) aproxima-se da medida Silhueta Nítida (do inglês, *crisp silhouette)* (SN) que serve de base para definir a SF (CAMPELLO; HRUSCHKA, 2006). A SN é a contraparte *rígida* da SF e é usada para algoritmos de agrupamento *rígido* aplicados a conjuntos de dados rígidos em que não há sobreposição entre os agrupamentos. Por outro lado, aumentar λ afasta SF de SN diminuindo a importância relativa dos objetos de dados em áreas sobrepostas. Assim, aumentar λ tende a enfatizar o efeito de revelar regiões menores com densidades de dados mais altas (subclusters), se existirem. Tal efeito pode ser particularmente útil, por exemplo, ao lidar com conjuntos de dados com ruídos.

SF é um IVC de maximização. Portanto, quanto maior o valor de SF para um determinado valor de c, melhor é esse agrupamento específico. O objetivo ao usar SF para avaliar a configuração de clusters gerada pelo FCM é encontrar o valor de c mais apropriado para organização dos dados.

Definidas as normas para cálculo da distância entre as instâncias e o método para avaliação da quantidade de grupos, a solução está pronta para ser utilizada com o dataset coletado utilizando as ferramentas descritas na seção 4.3. A Figura 4.23 ilustra a solução completa para o aprendizado da política de seleção de *setplays*.

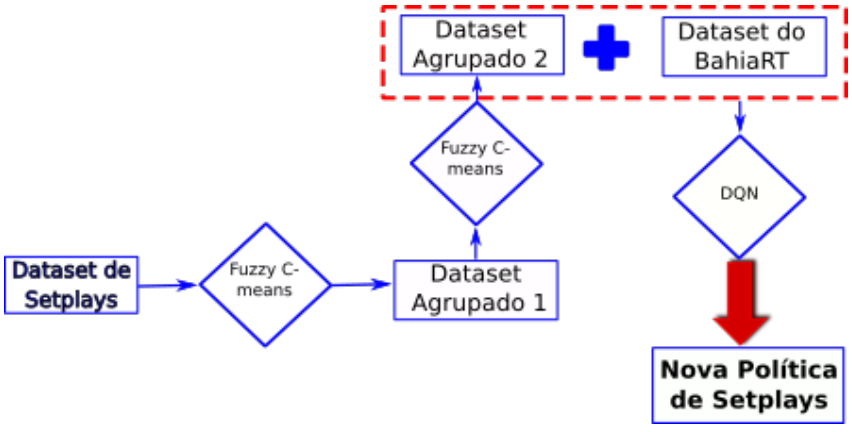


Figura 4.23: Solução completa para o aprendizado de uma política de *setplays*. As demonstrações são agrupadas após duas passagens do FCM gerando o dataset agrupado2. Juntamente com espaço de observação coletado no dataset do BahiaRT, o DQN aprende uma política de seleção de *setplays*.

O Dataset do BahiaRT irá usar informações obtidas através das percepções dos agentes do SMA para compor o espaço de obervações do DQN. Este procedimento será explicado na próxima seção.

4.5 APRENDENDO A USAR SETPLAYS COM BASE EM DEMONSTRAÇÕES

A etapa final da solução apresentada nesta tese consiste em treinar o SMA BahiaRT para aprender, a partir do dataset organizado em grupos, uma política de seleção de *setplays* que possa ser utilizada nas partidas de futebol de robôs na 3DSSIM.

O FSF utiliza um gerenciador de *setplays* baseado na abordagem de Raciocínio Baseado em Casos (RBC) (WANGENHEIM; WANGENHEIM, 2003) (ROS et al., 2009). Esta abordagem constrói um histórico de casos a partir da aplicação dos *setplays* disponíveis para os agentes do time. O gerenciador de *setplays* armazena informações sobre a execução de cada *setplay* e o sucesso ou falha correspondente. Se em determinado momento do jogo um ou mais *setplays* forem viáveis, este sistema selecionará o *setplay* com melhor avaliação. A avaliação depende do sucesso prévio do *setplay*, tanto no jogo atual quanto nos anteriores. Embora, seja uma solução já utilizada anteriormente com o FSF (MOTA; REIS; LAU, 2011), não é escalável para um grande conjunto de *setplays* como o que é utilizado neste trabalho.

Esta foi a motivação para a organização do dataset em grupos. A estratégia adotada nesta tese irá aplicar o gerenciador de *setplays* disponível no FSF com a solução de RBC apenas dentro de cada grupo, ou seja, considerando apenas os *setplays* dentro de um grupo. Para que isto seja possível, é necessário ter uma política de seleção do grupo que seja mais útil em cada situação de jogo. A Figura 4.24 ilustra a arquitetura do BahiaRT após a aprendizagem da política de seleção de *setplays*.

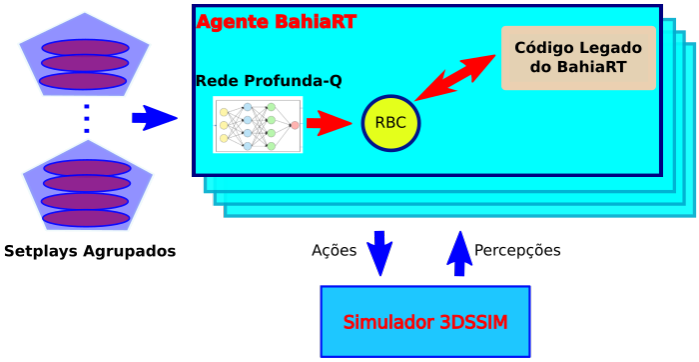


Figura 4.24: Arquitetura do BahiaRT após o aprendizado da política de seleção de *setplays* a partir do conjunto de demonstrações coletadas e organizadas em grupos.

Para melhor caracterizar cada situação de jogo, todas as posições de todos os jogadores em campo e posição da bola, dentre outras variáveis precisam ser consideradas. Além disto, com uma grande quantidade de *setplays* a escolher, uma solução capaz de lidar com este volume de informações é necessária. Estas características levaram à escolha do DQN como solução para aprender a política de seleção dos grupos do dataset que serão usados em cada situação de jogo.

Para modelar o problema, o espaço de observação O será definido pelos valores das propriedades descritas na Tabela 4.4. A cada ciclo de simulação, a posição de cada jogador em campo dos dois times, a posição da bola e o modo de jogo são atualizados, gerando potencialmente um novo estado.

Tabela 4.4: Propriedades que compõem os estados no espaço de observação dos agentes.

Propriedade	**Descrição**	**Tipo de dado**
$(x_{t1}, y_{t1}), \ldots, (x_{t11}, y_{t11})$	Posições dos 11 jogadores do BahiaRT no instante atual.	Pares de coordenadas reais
$(x_{o1}, y_{o1}), \ldots, (x_{o11}, y_{o11})$	Posições dos 11 jogadores do time oponente no instante atual.	Pares de coordenadas reais
(x_b, y_b)	Posição da bola no instante atual	Par de coordenadas reais
modo de jogo	Indica a situação atual da partida (em andamento, tiro de meta, escanteio, etc)	Inteiro
zona do campo	indica a zona do campo onde a bola se encontra.	Inteiro $\in \{0, \ldots, 35\}$

O espaço de ações A é definido por um número inteiro que representa a identificação do grupo de *setplays* gerado pelo organizador do dataset, conforme definido a seguir:

$$A = \{1, \ldots, C^*\}, \tag{4.8}$$

onde C^* é a quantidade total de grupos encontrada pelo organizador *fuzzy* do dataset definido na seção 4.4.

Os episódios podem ser iniciados com valores quaisquer das propriedades na Tabela 4.4. O término do episódio pode acontecer caso uma das seguintes condições seja satisfeita:

- um *setplay* for executado e concluído com sucesso;
- um *setplay* for executado e concluído com falha;
- um gol for marcado;
- um gol for sofrido;
- bola for jogada para fora do campo;
- o intervalo de tempo decorrido desde o início do episódio atingir um limite $\bar{t}$, sem que nenhuma das condições anteriores seja satisfeita.

Ao final de cada episódio, é calculada uma função de recompensa definida por

$$r(s_i, a_i) = \frac{\Delta x_B}{|\Delta x_B|} \times 2^{|\Delta x_B|} \times \left[1 + 10 \times (\text{flag}_{\text{GM}} + \text{flag}_{\text{GS}}) + 3 \times \text{flag}_{\text{SS}} + \frac{\text{flag}_{\text{SF}}}{2}\right], \tag{4.9}$$

onde $s_i \in O$ é o estado observado no instante i e $a_i \in A$ é a ação escolhida pelo agente no instante i. $\Delta x_B = x_B^f - X_B^s$, onde x_B^f e X_B^s são a coordenada no eixo x da bola no instante final e inicial do episódio, respectivamente. As flags são definidas por

- $\text{flag}_{\text{GM}} = 1$, se um gol foi marcado no episódio, e 0 caso contrário;
- $\text{flag}_{\text{GS}} = 1$, se um gol foi sofrido no episódio, e 0 caso contrário;
- $\text{flag}_{\text{SS}} = 1$, se um *setplay* foi concluído com sucesso no episódio, e 0 caso contrário;
- $\text{flag}_{\text{SF}} = 1$, se um *setplay* foi concluído com falha no episódio, e 0 caso contrário;

A função de recompensa foi construída para fornecer um viés ao treinamento que priorize o deslocamento da bola no eixo x, ou seja, levar a bola para o campo de ataque será sempre recompensando positivamente. Quanto mais a bola se desloca positivamente no eixo x, maior a recompensa positiva. Da mesma forma, se o deslocamento for negativo, ou seja, a bola mover-se em direção à defesa, a recompensa será negativa. Assim, mesmo que os *setplays* não consigam sempre resultar em gols marcados ou sofridos, eles devem reduzir a probabilidade de sofrer gols e aumentar a probabilidade de marcar gols ao buscar manter a bola majoritariamente no campo de ataque. A recompensa atribui um bônus de 10 vezes a recompensa pelo deslocamento quando o episódio resultar num gol marcado e, na mesma proporção, atribui recompensa negativa quando um gol é sofrido no episódio. A recompensa contabiliza um bônus de três vezes a recompensa pelo deslocamento sempre que um *setplay* é concluído com sucesso. Também é contabilizada uma punição equivalente à metade da recompensa pelo deslocamento, sempre que um *setplay* for concluído com falha.

Então, a função de recompensa busca dar o viés do treinamento para que a política de seleção de grupos de *setplays* aprendida busque encontrar grupos para cada situação que potencialize as chances de executar os *setplays* com sucesso, marcando gols, levando a bola ao ataque e impedindo gols e retrocessos da bola em direção à defesa.

Para dar suporte ao treinamento, é utilizado o framework Open AI Gym (BROCKMAN et al., 2016). O OpenAI Gym é um kit de ferramentas para pesquisa de aprendizado por reforço. Inclui uma coleção crescente de problemas de *benchmark* que expõem uma interface comum e um site onde as pessoas podem compartilhar seus resultados e comparar o desempenho dos algoritmos.

O OpenAI Gym traz uma série de ambientes simulados que servem de suporte ao treinamento de políticas de controle utilizando aprendizagem por reforço. Entretanto, não há um ambiente para o simulador da 3DSSIM. Alguns trabalhos desenvolveram ambientes próprios, mas específicos para os códigos dos agentes dos SMA utilizados (KASAEI et al., 2021)(ABREU et al., 2021). Além disso, estes ambientes não estão disponíveis e, mesmo se estivessem, não seriam compatíveis com outros SMA a exemplo do BahiaRT.

Nesta tese, foi desenvolvido o BahiaRT Gym[8], um ambiente completamente compatível com o OpenAI Gym, tornando fácil o treinamento utilizando aprendizagem por reforço com o ambiente da 3DSSIM. O BahiaRT Gym tem uma arquitetura desacoplada do SMA BahiaRT e pode ser utilizado por qualquer grupo de pesquisa ou equipe de futebol de robôs que pretenda utilizar o simulador 3DSSIM em seu trabalho. A Figura 4.25 apresenta a arquitetura do projetada para o BahiaRT Gym.

[8]https://bitbucket.org/bahiart3d/bahiart-gym

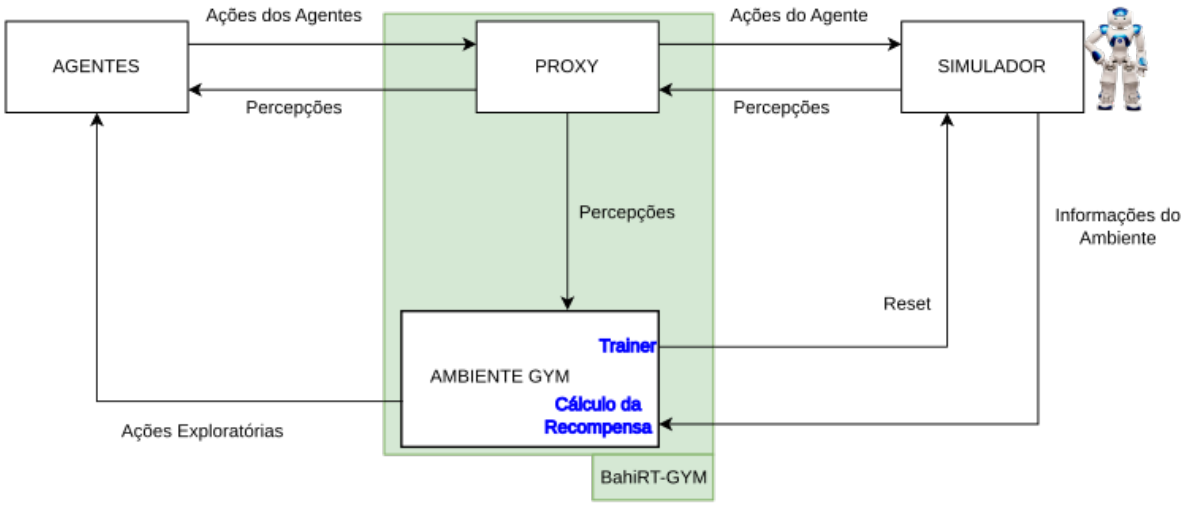


Figura 4.25: Arquitetura do BahiaRT Gym é desacoplada do time BahiaRT permitindo seu uso por qualquer pesquisador que deseje realizar trabalhos de aprendizagem por reforço com o simulador da 3DSSIM.

O BahiaRT Gym tem dois elementos centrais: Proxy e Ambiente GYM. O Proxy é responsável por interceptar toda comunicação entre os agentes que integram o SMA e o simulador 3DSSIM. Através desta interceptação, o Proxy consegue obter todas as percepções enviadas pelo simulador aos agentes, bem como, as ações enviadas pelos agentes ao simulador. Estes dados são fornecidos ao Ambiente Gym que implementa a interface do OpenAI Gym.

O ambiente GYM utiliza as informações obtidas pelo Proxy e as informações do ambiente coletadas diretamente com o simulador para calcular recompensas e atualizar o estado do ambiente. O Ambiente GYM utiliza a interface *Reset* do OpenAI Gym para iniciar um novo episódio. Para isto, ele utilizará os comandos do modo *Trainer* aceitos pelo simulador na porta destinada à conexão do RoboViz. Por este motivo, o Ambiente GYM conecta no simulador da mesma forma que o RoboViz o faz. Assim, além de poder usar os comandos do modo *Trainer*, as informações do ambiente obtidas por este canal são exatas e livres de ruídos, ao contrário das percepções que o simulador envia para os agentes. Utilizando os comandos do modo *Trainer*, é possível configurar o início do episódio da forma exata que tenha sido planejado nos experimentos. Por exemplo, é possível posicionar cada robô numa posição específica, posicionar a bola na posição desejada, definir o modo de jogo necessário ao episódio, entre outras opções.

O BahiaRT Gym também tem suporte para múltiplos agentes, ou seja, é possível realizar treinamentos com todos os jogadores em campo, inclusive em partidas contra times adversários completos. Este é um diferencial que a maioria dos ambientes utilizados para o simulador da 3DSSIM não apresenta. Não foi encontrado na literatura trabalhos que utilizem o OpenAI Gym com o simulador da 3DSSIM com mais de um agente em campo. Em geral, utilizam o ambiente Gym para aprendizagem de habilidades individuais de jogadores (ex: chute, caminhada, corrida, etc) que não necessitam de mais que um

robô em campo durante o treinamento.

O BahiaRT Gym acompanha um ambiente de demonstração que mostra na prática para o usuário como usar a ferramenta para criar seus próprios ambientes de treinamento.

Esta é, portanto, mais uma contribuição desta tese para a comunidade científica que atua com aprendizagem por reforço, em particular no ambiente da 3DSSIM.

Além do BahiaRT Gym, este trabalho utiliza implementação do algoritmo DQN disponível na biblioteca *Stable Baselines 3* (RAFFIN et al., 2021). É uma implementação eficiente computacionalmente e parametrizável, permitindo experimentar uma variedade de hiper-parâmetros para o algoritmo escolhido. A Figura 4.26 ilustra a arquitetura do treinamento da rede profunda Q que permite que o BahiaRT tenha uma política geral para seleção de *setplays*, a partir da base disponível com demonstrações de *setplays*.

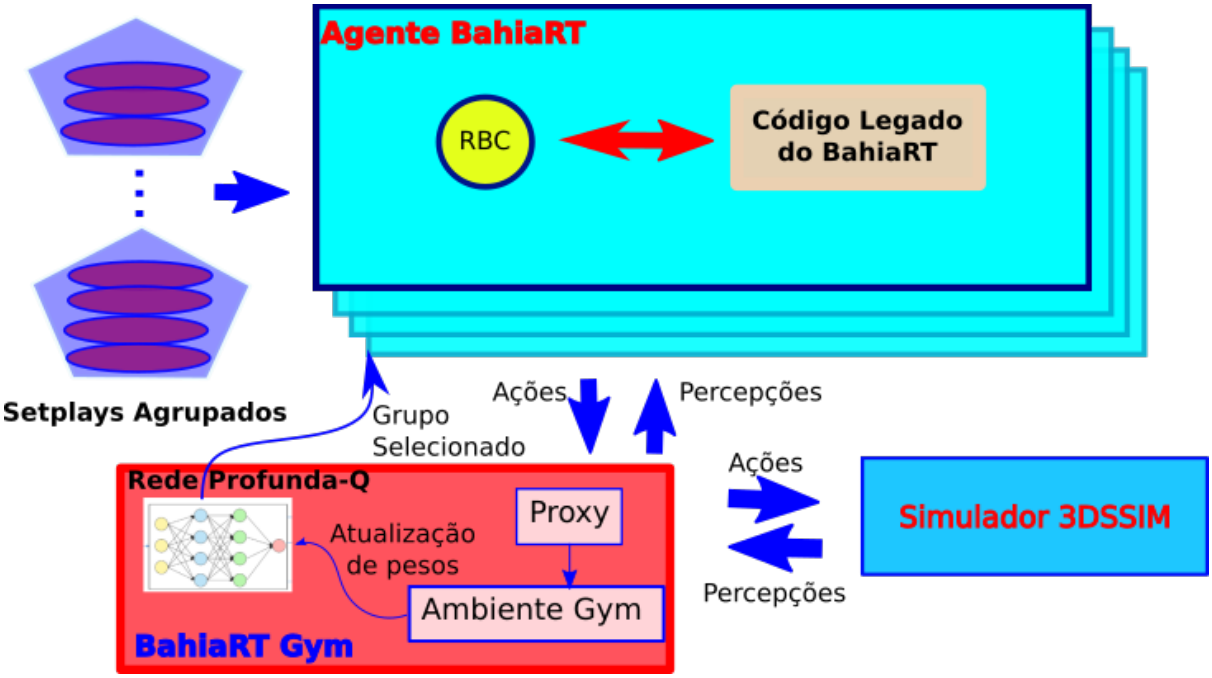


Figura 4.26: Arquitetura de treinamento da Rede Q Profunda que será usada pelo BahiaRT. O BahiaRT Gym é responsável por calcular as recompensas e atualizar os pesos da rede a cada episódio de treinamento.

4.6 CONSIDERAÇÕES FINAIS

Neste capítulo, foi apresentada toda a metodologia para o desenvolvimento da solução apresentada nesta tese. Ficam evidenciados os diversos produtos gerados durante o desenvolvimento do trabalho:

- Nova versão do RoboViz, incluindo o modo de demonstração que permite capturar situações de jogo para iniciar novas demonstrações;
- Nova versão do SPlanner que permite iniciar e gerar uma demonstração de *setplay*

a partir da situação capturada no RoboViz, além de incluir comportamentos e elementos ausentes na versão anterior da ferramenta;

- BahiaRT Setplays Toolkit: kit de ferramentas que reúne em um container docker todas as ferramentas e dados necessários para qualquer pessoa poder criar e enviar demonstrações, permitindo o uso da estratégia de *crowdsourcing*;

- Organização das demonstrações em um dataset organizado em clusters em dois níveis, gerando o dataset agrupado além da solução modificada do FCM com novas normas de distância para suportar tipos de dados não escalares;

- BahiaRT Gym: ambiente baseado on OpenAI Gym que integra o simulador da 3DSSIM de forma desacoplada do BahiaRT, permitindo o uso da aprendizagem por reforço em problemas diversos usando o ambiente da 3DSSIM;

- Modelo de aprendizagem por reforço para uso com o algoritmo DQN para aprender uma política geral para seleção de *setplays*.

Todos estes produtos evidenciam as contribuições desta tese para o avanço do estado da arte. No próximo capítulo, os experimentos e resultados dos processos de organização do dataset de demonstrações e da aprendizagem da política de seleção de *setplays* são apresentados.

Capítulo

5

Neste capítulo, os experimentos realizados para organização do dataset e aprendizagem da política de setplays com seus respectivos resultados são apresentados.

RESULTADOS EXPERIMENTAIS

A estratégia de *crowdsourcing* utilizada (ver Capítulo 4) gerou um total de 382 demonstrações de *setplays*. Na Seção 5.1, os experimentos e resultados realizados para definir os melhores parâmetros para a partição Fuzzy utilizada na organização do dataset de demonstrações obtido são apresentados. Na Seção 5.2, os experimentos e resultados realizados utilizando o ambiente descrito no capítulo anterior para treinamento de uma rede Q capaz de generalizar uma política de seleção de *setplays* são apresentados. A solução apresentada integra o mecanismo pré-existente no *FCPortugal Setplays Framework* (FSF) utilizando Raciocínio Baseado em Casos (RBC).

5.1 ORGANIZAÇÃO DO CONJUNTO DE DEMONSTRAÇÕES

Um dos grandes desafios no uso das soluções de agrupamento é a definição da quantidade de grupos (clusters) adequada para a organização das instâncias do conjunto de dados. Usualmente, a definição da quantidade apropriada de clusters em um processo de agrupamento Fuzzy consiste em executar o algoritmo de agrupamento para diferentes números de clusters, avaliando cada agrupamento por meio de um Índice de Validação de Clusters (IVC). Assim, a quantidade de clusters que fornecer o melhor valor de IVC será escolhida. Neste trabalho o IVC utilizado foi a Silhueta Fuzzy (SF), conforme apresentado na Seção 4.4.

Além da quantidade de clusters nos dois níveis da solução descrita na seção 4.4, alguns parâmetros para o *Fuzzy C-Means* (FCM) e o cálculo da SF precisam ser definidos empiricamente. O fator de fuzzificação m, utilizado nas Equações (3.3) e (3.4), e o coeficiente de ponderação λ, utilizado no cálculo da SF, precisam ser definidos empiricamente. Estes dois parâmetros são importantes para encontrar uma configuração de clusters que seja adequada ao nível de difusão do conjunto de dados em questão.

Na busca pela melhor organização dos *setplays* obtidos, os experimentos apresentados nesta seção foram divididos em dois grupos. No primeiro, um conjunto menor de demonstrações foi utilizado, antes da coleta do conjunto de dados final, para definir os valores

de m e λ que melhor exploram o nível de sobreposição entre clusters para a natureza dos dados que compõem o dataset. No segundo grupo, o conjunto completo foi utilizado. Por ser mais completo, este conjunto demanda mais recurso computacional, mas é uma fase importante para que as demonstrações de *setplays* sejam utilizadas como planejado na solução de aprendizagem por reforço que aprende a política de seleção de *setplays*.

No primeiro grupo de experimentos, um conjunto preliminar de estudos com 18 instâncias permitiu definir, experimentalmente, os valores de m e λ que sugerem obter maiores valores de SF. Foram testados valores de $m = 1,5$ e $m = 2$. Já para λ foram testados os valores 1 e 2. Os resultados demonstram que os valores $m = 2$ e $\lambda = 2$ são apropriados para o conjunto de dados utilizado, uma vez que o máximo valor de SF foi obtido para esta combinação de valores, quando o agrupamento foi realizado para $C^{(1)} = 3$ grupos. A Figura 5.1 ilustra a variação da SF para os valores de $2 \leq C^{(1)} \leq 8$ resultantes destes experimentos (SIMõES; SILVA; NOGUEIRA, 2020).

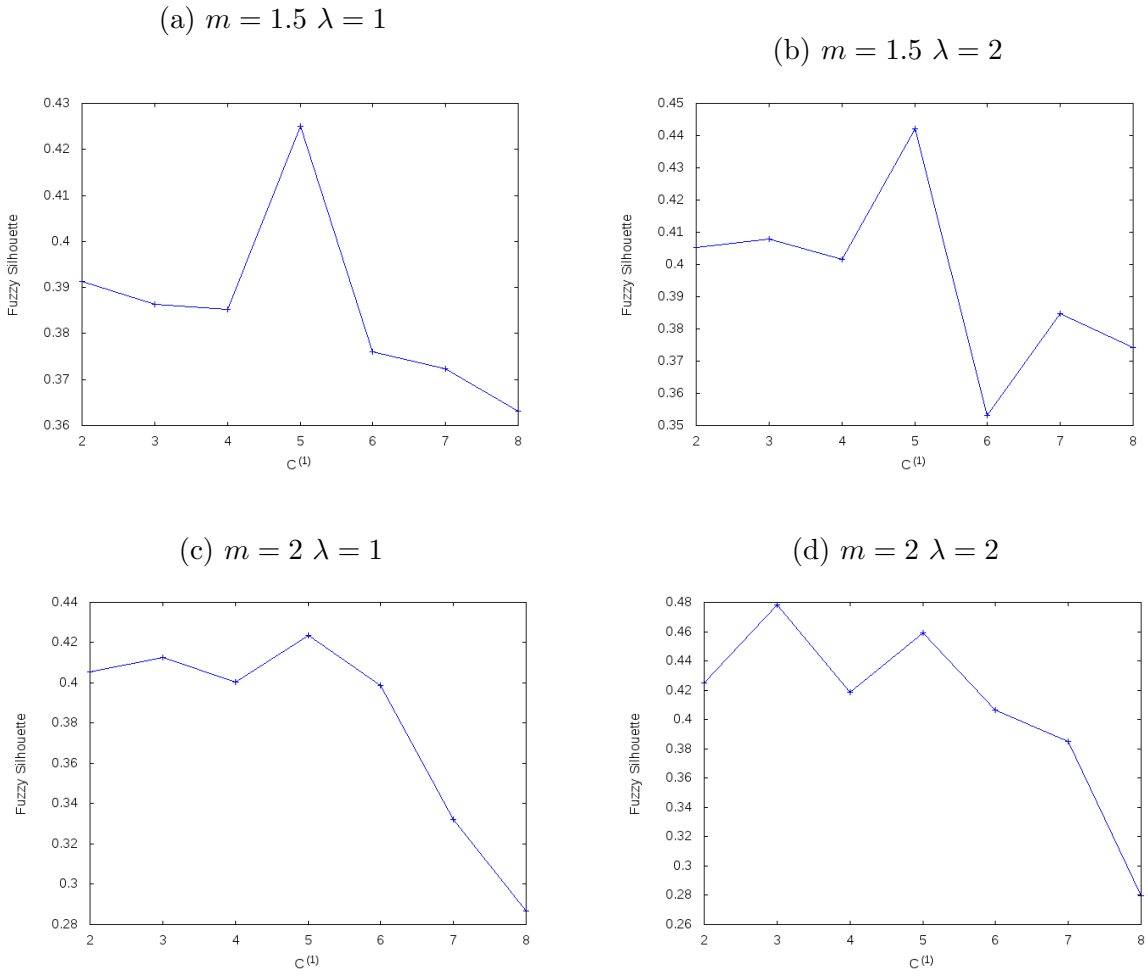


Figura 5.1: Variação da SF para diferentes valores de m e λ aplicados em um conjunto preliminar de 18 demonstrações.

Em seguida, os melhores valores de m e λ obtidos foram testados com um conjunto de dados maior, composto por 181 instâncias (SIMõES et al., 2021). Este ainda não é o conjunto completo de demonstrações, mas já amplia o espaço de validação dos parâmetros.

Para obter resultados mais confiáveis acerca da quantidade apropriada de clusters para o conjunto de demonstrações de *setplays*, o algoritmo de agrupamento FCM foi executado 10 vezes, em função da inicialização aleatória, variando a quantidade de clusters $C^{(1)}$ no intervalo $[\frac{\sqrt{n}}{2}, 2 \times \sqrt{n}]$, considerando n a quantidade de instâncias a serem agrupadas. A cada execução, o agrupamento obtido foi avaliado por meio da SF. O valor da SF obtido por cada quantidade de grupos é o valor médio obtido após as 10 iterações. Ao executar este experimento o maior valor de SF foi obtido para o último valor $C^{(1)} = 27$. Para não restar dúvida se os valores de $C^{(1)} > 27$ apresentariam uma tendência decrescente para SF, estendeu-se o valor de $C^{(1)}$ até 30 para este experimento. A Figura 5.2 exibe os resultados dos experimentos. A variação da SF para $6 \leq C^{(1)} \leq 30$ é apresentada na Figura 5.2a.

(a) Variação da SF no nível 1

(b) Quantidade de instâncias por grupo $c^{(1)}$ para $1 \leq c^{(1)} \leq 27$.

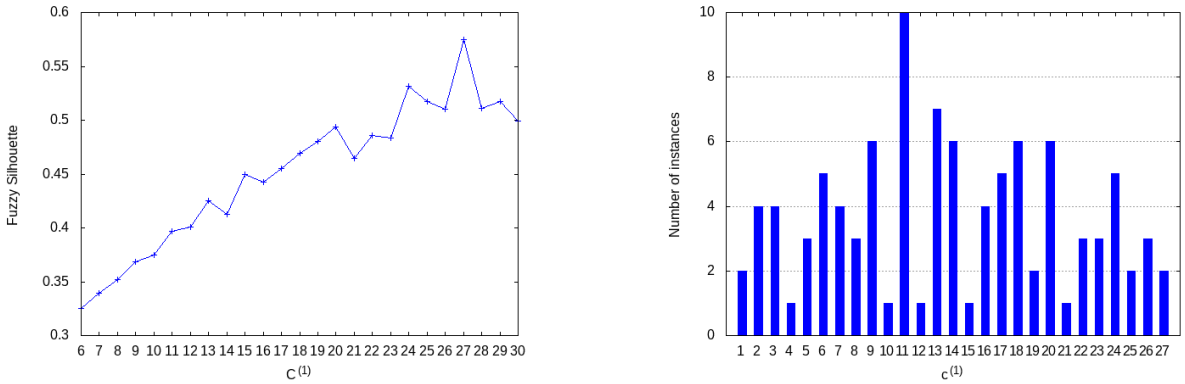


Figura 5.2: Resultados dos experimentos para um conjunto de $n = 181$ instâncias com $6 \leq C^{(1)} \leq 30$.

O maior valor obtido pelo IVC foi $SF = 0,57$ para $C^{(1)} = 27$ clusters. Ficou comprovado que, para os valores $C^{(1)} > 27$ próximos a $C^{(27)}$, os valores de SF assumem uma tendência decrescente. Isto confirma o número de clusters $C^{(1)} = 27$ como melhor escolha para o nível 1. Então, este conjunto de 181 demonstrações de *setplays* tem sua melhor organização dividido em 27 grupos. A Figura 5.2b mostra a distribuição da quantidade de instâncias pelos 27 grupos. É possível observar que a quantidade máxima de instâncias em um determinado grupo, na Figura representada pela número 11, é igual a 10. A maior parte dos grupos tem entre 2 e 6 instâncias. Esta quantidade razoavelmente pequena de *setplays* por grupo é favorável ao uso desta organização para o ambiente da *RoboCup 3D Soccer Simulation* (3DSSIM), pois durante a simulação os agentes geram novas ações obedecendo a um período de simulação de $20ms$. Portanto, se a escolha dos *setplays* precisar analisar um grande número de instâncias, pode se tornar inviável em termos de

custo computacional. Os requisitos de tempo presentes no ambiente de simulação não permitem longos tempos de deliberação por parte dos agentes.

Seguindo a solução descrita no capítulo anterior, o procedimento é replicado em cada um dos 27 grupos para gerar a subdivisão de grupos em segundo nível. Cada grupo foi reagrupado em até 7 clusters pelo algoritmo FCM executado no nível 2. Entretanto, os grupos que possuem apenas uma instância não podem ser subdivididos.

Os resultados destes experimentos iniciais forneceram o aprendizado necessário para os experimentos finais, os quais utilizam o dataset com as 382 demonstrações de *setplays* obtidas. O procedimento aplicado consiste em executar 10 iterações do FCM para obter os melhores valores de SF para cada valor de $C^{(1)}$. Foram considerados os parâmetros $m = 2$ e $\lambda = 2$. A Figura 5.3 exibe os resultados obtidos.

(a) Variação da SF no nível 1

(b) Quantidade de instâncias por grupo $c^{(1)}$ para $1 \leq c^{(1)} \leq 10$.

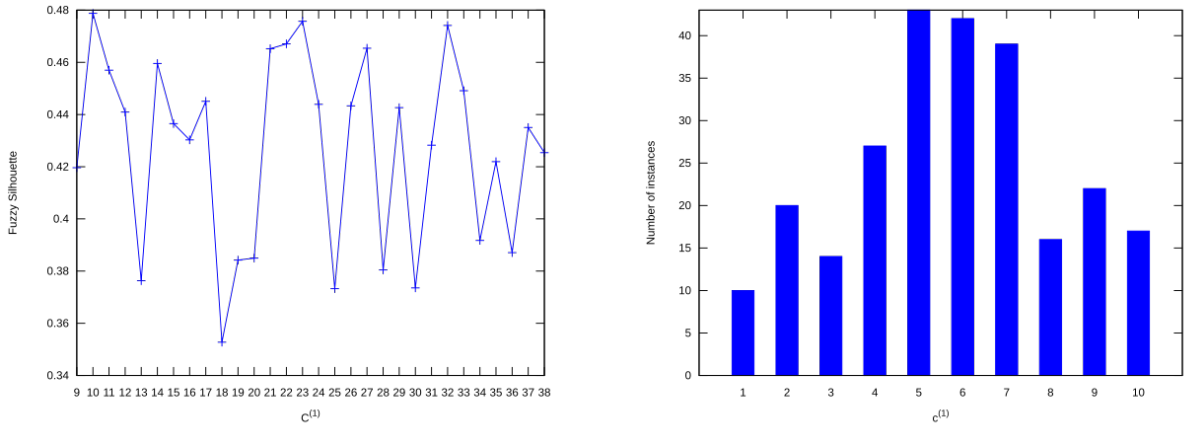


Figura 5.3: Resultados dos experimentos para um conjunto de $n = 382$ instâncias com $9 \leq C^{(1)} \leq 38$.

Na Figura 5.3a, é possível observar a variação da SF para as quantidades de grupos testadas $9 \leq C^{(1)} \leq 38$, considerando o intervalo de valores estabelecidos anteriormente. Observa-se uma variação grande da SF entre os grupos, refletindo a maior diversidade de *setplays* originária da abordagem *crowdsourcing*. Ao mesmo tempo, percebe-se alguns arranjos de grupos que mostram-se com valores destacados de SF, como por exemplo $C^{(1)} = 10$, $C^{(1)} = 23$ e $C(1) = 33$.

Na Figura 5.3b pode-se observar a distribuição da quantidade de instância por grupo, quando toma-se o valor de $C^{(1)} = 10$ que obteve a maior SF. Percebe-se $10 \leq n_{c^{(1)}} \leq 42$, mas a média é de 25 instâncias por grupo. Estes valores evidenciam a necessidade do segundo nível de organização. Trabalhar com grupos deste tamanho pode inviabilizar o processo de seleção dos *setplays* usando a solução de RBC existente no FSF.

Para a execução do segundo nível, o mesmo procedimento foi adotado: para cada valor de $1 \leq c^{(1)} \leq 10$, são executadas 10 iterações do FCM com valores de $\frac{\sqrt{n_{c^{(1)}}}}{2} \leq$

$C^{(2)} \leq 2 \times \sqrt{n_{c^{(1)}}}$. A Figura 5.4 exibe os resultados do segundo nível.

(a) Número de clusters no nível 2($C^{(2)}$) em que cada grupo do nível 1($c^{(1)}$) será desmembrado .

(b) Valor máximo da SF para cada grupo $c^{(1)}$ após a execução do agrupamento no nível 2.

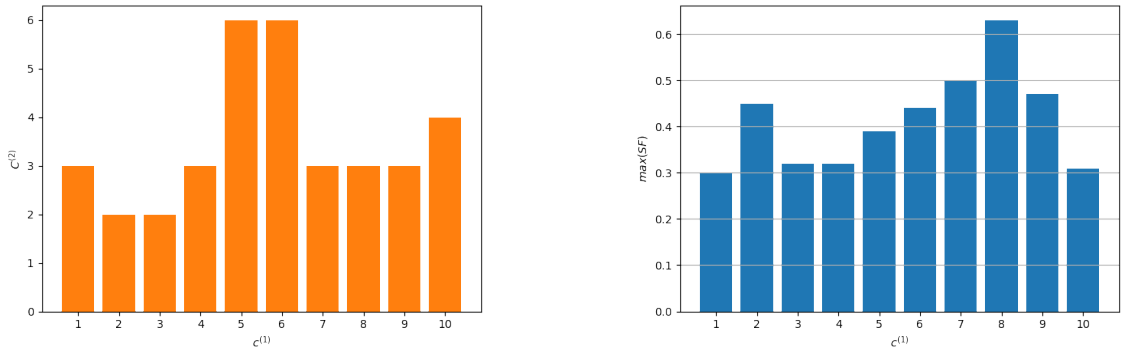


Figura 5.4: Resultados dos experimentos para o agrupamento do nível 2 em um conjunto de $n = 382$ instâncias organizado em 10 grupos no nível 1.

Os 10 grupos do nível 1 poderão ser divididos em 2, 3, 4 ou 6 grupos no nível 2, conforme ilustrado na Figura 5.4a. Esta quantidade de clusters é resultante no valor máximo da SF obtido pelo agrupamento do nível 2 realizado em cada grupo $1 \leq c^{(1)} \leq 10$.

A Figura 5.4b exibe os valores máximos de SF obtidos após a partição fuzzy gerada pelo agrupamento realizado a partir de cada $1 \leq c^{(1)} \leq 10$. Este resultado gera um total de 35 grupos após dois níveis de agrupamento. Estes grupos irão compor o espaço de ações A do mecanismo de aprendizagem por reforço usado para treinamento da política de seleção de *setplays*. A próxima seção apresenta os resultados obtidos neste treinamento.

5.2 POLÍTICA DE SELEÇÃO DE SETPLAYS

Para validar o modelo de aprendizagem por reforço, descrito no capítulo anterior, foram utilizados os 35 grupos de demonstrações gerados pelo organizador de dataset, conforme resultados da seção anterior.

O espaço de ações do modelo foi definido como $A = 1, \ldots, 35$ onde cada $c^{(2)} \in A$ representa uma possível decisão da política de seleção de *setplays*. Quando um dos grupos é selecionado pela rede Q, o mecanismo de RBC do FSF entra em ação utilizando apenas os *setplays* que integram o grupo selecionado.

Para executar o treinamento da rede, foram escolhidas três equipes oponentes, baseando-se no histórico de confrontos com a equipe *Bahia Robotics Team* (BahiaRT) em competições e treinamentos anteriores. Foi selecionada uma equipe de nível inferior, que costuma perder ou empatar com o BahiaRT, WITS-FC[1]; outra equipe de mesmo nível,

[1]Equipe da Wits University, da África do Sul. https://www.wits.ac.za/

que costuma ter mais empates e resultados equilibrados com triunfo para um dos lados por um gol de diferença, ITAndroids[2]; e uma equipe de nível superior, que costuma vencer todos os confrontos com o BahiaRT por placares com grande diferença de gols, magmaOffenburg[3].

O treinamento foi executado sobre 100 jogos contra cada equipe, utilizando o ambiente do BahiaRT Gym conectado para controlar os episódios. A função *reset()* do Gym é usada para reiniciar os episódios. Neste caso, como o treinamento acontece com jogos contínuos, a cada iteração, esta função verifica se uma das condições apresentadas na seção 4.5 para o fim do episódio foi atingida. A função *reset()* informa à função *step()*, também parte da API do Gym, para que ela cuide do cálculo da recompensa e início do episódio seguinte. A função *reset()* ainda é responsável por monitorar o fim das partidas, acionando a função do BahiaRT Gym responsável por reiniciar nova instância do simulador para continuar o treinamento em uma nova partida.

As informações coletadas do simulador e dos agentes são usadas para o cálculo da recompensa conforme a equação (4.9). A recompensa é utilizada para atualização dos pesos da rede Q utilizando a implementação do algoritmo *Deep Q Network* (DQN) do *Stable Baselines* 3 (RAFFIN et al., 2021). Para este treinamento, os parâmetros *default* do *Stable Baselines* foram mantidos.

Para reduzir as chances de sobreajuste do treinamento, as partidas contra os diferentes adversários foram executadas de forma alternada, pois a rede que está sendo treinada é a mesma, independente do adversário. Então, se fossem executadas 100 partidas para contra um mesmo adversário, quando o treinamento do adversário seguinte iniciasse, a rede já teria um viés relacionado aos posicionamentos e comportamentos do adversário anterior.

O time BahiaRT ganhou um modo de treinamento que o faz conectar-se ao BahiaRT Gym para receber as indicações de ações exploratórias gerada pela rede Q através do BahiaRT Gym, enquanto o treinamento estiver sendo executado. O BahiaRT retorna informações necessárias, de acordo com o treinamento sendo executado. Neste treinamento, o agente líder, no passo final do *setplay* executado, informa o valor da flag_{SS}, indicando se um *setplay* foi concluído com sucesso no episódio, e da flag_{SF}, indicando se um *setplay* foi concluído com falha no episódio. Esses valores são usados para o cálculo da recompensa, conforme Equação 4.9 apresentada no capítulo anterior.

Quando o agente do BahiaRT não está em plena execução de um *setplay*, se ele recebe uma ação do BahiaRT Gym indicando um grupo de *setplays* selecionado, ele irá iniciar o gerenciador de *setplays* do FSF para escolher um *setplay* através da estratégia de RBC. É possível que nenhum dos *setplays* seja selecionado e neste caso não há início do mesmo. Quando o agente não recebe uma ação exploratória, nem tem um *setplay* apto a ser iniciado, ele realiza o comportamento reativo padrão que o time já utilizava antes deste trabalho.

Com este método de treinamento, foram executadas as 300 partidas gerando uma política de seleção de *setplays* aprendida. A avaliação da política se deu com testes

[2]Equipe do Instituto Tecnológico da Aeronáutica (ITA-SP). http://www.itandroids.com.br/

[3]Equipe da Hochschule Offenburg University, da Alemanha. https://robocup.hs-offenburg.de/

do BahiaRT, fora do modo de treinamento, utilizando a política aprendida para jogar mais 100 partidas contra cada um dos três times adversários. Nesta etapa, não há mais interação com o BahiaRT Gym, os agentes carregam para a memória o modelo da rede Q treinada e a utilizam a cada ciclo de simulação para escolha de um grupo de *setplays* para uso.

Para avaliação dos resultados, o desempenho global do BahiaRT contra os três adversários selecionados foi considerado. Para comparação, executou-se 100 partidas com cada adversário contra o time BahiaRT anterior, sem usar a política de seleção de *setplays* nem a base de demonstrações de *setplays* construída neste trabalho. Os resultados obtidos a partir desses testes são apresentados na Tabela 5.1. Outras 100 partidas contra cada adversário foram executadas contra o time BahiaRT atualizado com o dataset de demonstrações e a política de seleção de *setplays* treinada. Os resultados obtidos a partir desses testes são apresentados na Tabela 5.2. A análise dos resultados se dá pelas quantidades de triunfos, empates e derrotas, além das médias de gols marcados (GM) e gols sofridos (GS).

Tabela 5.1: Jogos realizados sem aprendizagem de setplays

Times adversários	Triunfos	Empates	Derrotas	GM		GS	
				Média	Desvio Padrão	Média	Desvio Padrão
ITAandroids	16	58	26	0,31	0,54	0,41	0,59
WITS_FC	59	41	0	0,87	0,90	0	0
magmaOffenburg	0	0	100	0,01	0,1	5,64	1,46

Tabela 5.2: Jogos realizados com aprendizagem de *setplays*

Times adversários	Triunfos	Empates	Derrotas	GM		GS	
				Média	Desvio Padrão	Média	Desvio Padrão
ITAandroids	26	53	21	0,45	0,73	0,28	0,45
WITS_FC	67	33	0	1,03	0,90	0	0
magmaOffenburg	0	0	100	0,01	0,10	5,14	0,97

Comparando os dois resultados, sem e com aprendizagem, observa-se que há uma variação. A quantidade de triunfos subiu de 16 para 26, com as correspondentes quedas na quantidade de empates e derrotas. Houve um aumento da média de $0,31$ gols por jogo para $0,45$ gols por jogo. Isto foi acompanhando de uma redução na quantidade de gols sofridos de $0,41$ para $0,28$.

A função de recompensa adotada no processo de aprendizagem de *setplays* atribui recompensas positivas pelo deslocamento da bola ao ataque, mesmo que os *setplays* não sejam concluídos com sucesso. Portanto, mesmo não gerando um aumento grande na média de gols marcados ou na redução dos gols sofridos, a manutenção da bola mais

tempo no ataque explica os resultados obtidos, gerando mais oportunidades de marcar gols e reduzindo os riscos de sofrer gols. O mapa de calor comparativo das partidas dos jogos antes e depois da aprendizagem de *setplays*, confirma esta hipótese, como pode ser visto na Figura 5.5. No mapa de calor o BahiaRT sempre defende o gol do lado esquerdo e ataca o gol do lado direito.

(a) BahiaRT (sem aprendizagem) x ITAndroids

(b) BahiaRT (com aprendizagem) x ITandroids

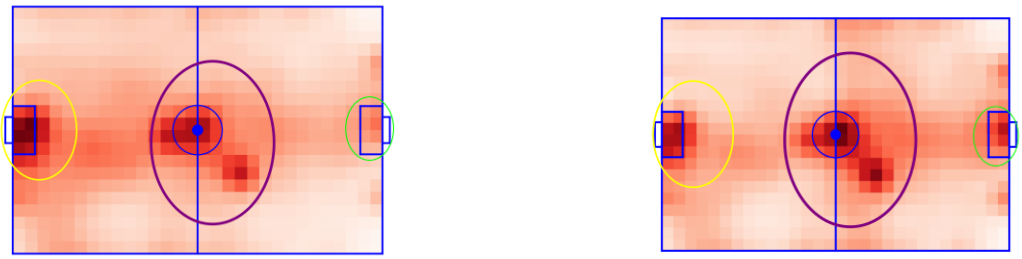


Figura 5.5: Mapa de calor da posição da bola e sua dispersão no campo durante o lote de 100 partidas do BahiaRT contra o ITAndroids sem e com a aprendizagem de *setplays*. No mapa de calor o BahiaRT sempre defende o gol do lado esquerdo e ataca o gol do lado direito.

A parte de baixo da figura sempre representa o lado direito do campo para o BahiaRT, enquanto a parte de cima é o lado esquerdo do campo. Quanto mais escura a tonalidade de vermelho, maior a frequência da bola naquela região durante as 100 partidas.

Observa-se nos mapas de calor dos jogos do BahiaRT contra o ITAndroids uma redução da presença da bola na área defensiva e um aumento desta frequência nas regiões de ataque, incluindo próximo ao gol adversário. Comparando-se as regiões no círculo amarelo nas Figuras 5.5a e 5.5b, fica clara a redução da concentração da posição da bola na área de defesa próxima ao gol defendido pelo BahiaRT (círculo amarelo). Isto explica a redução na quantidade de gols sofridos. Observa-se ainda um pequeno aumento na porção de ataque (círculos roxo e verde). O uso da política de *setplays*, permitiu levar mais a bola para o ataque, explicando o aumento na quantidade de gols marcados.

Analisando os resultados dos jogos do BahiaRT contra a equipe WITS-FC, percebe-se que não houve qualquer redução na superioridade de resultados do BahiaRT sobre esta equipe. Ao contrário, houve algum aumento no número de triunfos e redução nos empates. Nenhuma derrota continuou registrada, como pode ser visto na Tabela 5.2. A média de gols marcados teve um aumento, enquanto nenhum gol foi sofrido tanto pelo time sem os *setplays* aprendidos quanto pelo time com os *setplays* aprendidos. Avaliando a distribuição da posição da bola, pode-se observar a Figura 5.6

(a) BahiaRT (sem aprendizagem) x WITS-FC

(b) BahiaRT (com aprendizagem) x WITS-FC

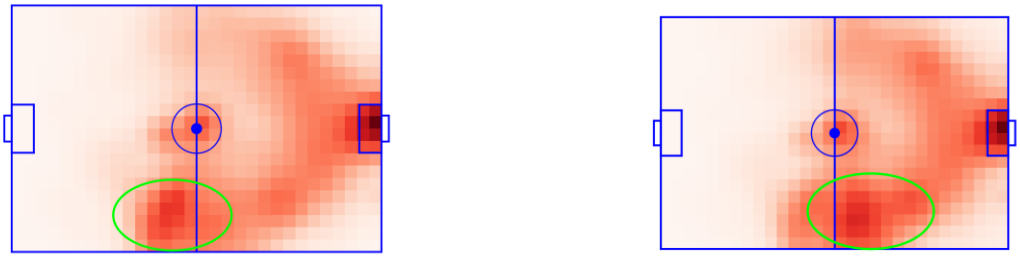


Figura 5.6: Mapa de calor da posição da bola e sua dispersão no campo durante o lote de 100 partidas do BahiaRT contra o WITS-FC sem e com a aprendizagem de *setplays*.

É possível observar o deslocamento da frequência da bola do meio campo defensivo para o ofensivo no lado direito BahiaRT(círculo verde) quando compara-se a Figura 5.6a com a Figura 5.6b. Esta diferença de regiões é suficiente para que o BahiaRT tente mais alguns chutes a gol de longa distância, com chances de ser bem sucedido. Este fato explica o aumento no número de gols marcados. O restante do mapa de calor, tem pouca variação, com destaque adicional para o aumento da frequência na área do gol adversário.

Por fim, a análise dos jogos contra o magmaOffenburg demonstra que não há uma mudança clara de desempenho global. O BahiaRT continua não conseguindo triunfar sobre este oponente mantendo a média de $0,01$ gols marcados por jogo. Mas percebe-se também uma diminuição na média de gols sofridos na Tabela 5.2. A variação dá-se dentro da faixa de desvio padrão, mas há indícios de uma redução neste volume de gols sofridos, possivelmente reduzindo os placares contra este adversário. A análise do mapa de calor na Figura 5.7 permite encontrar explicação para os resultados.

Observa-se que os jogos contra o magmaOffenburg costumavam concentrar-se fortemente no meio do campo e dentro da área defensiva do BahiaRT. Isto se explica pela estratégia de jogo da equipe alemã fortemente baseada em passes rápidos verticais em direção ao gol adversário. Isto faz com que a bola transite rapidamente do meio de campo para a área adversária até que a defesa seja vencida e ceda um gol. Mas na Figura 5.7b percebe-se uma maior dispersão da bola para as laterais do campo (retângulo roxo), ainda na metade defensiva do BahiaRT.

Possivelmente, o acionamento de *setplays* defensivos, através da nova política de *setplays*, dificultou um pouco mais a troca de passes do magmaOffenburg, levando-os a demorar mais de chegar na área adversária. De fato, quando comparado com a Figura 5.7a, percebe-se uma menor concentração na área defensiva do BahiaRT (círculo verde). Isto também explica a redução na média gols. Levando mais tempo para vencer a defesa

(a) BahiaRT (sem aprendizagem) x magma-Offenburg

(b) BahiaRT (com aprendizagem) x magma-Offenburg

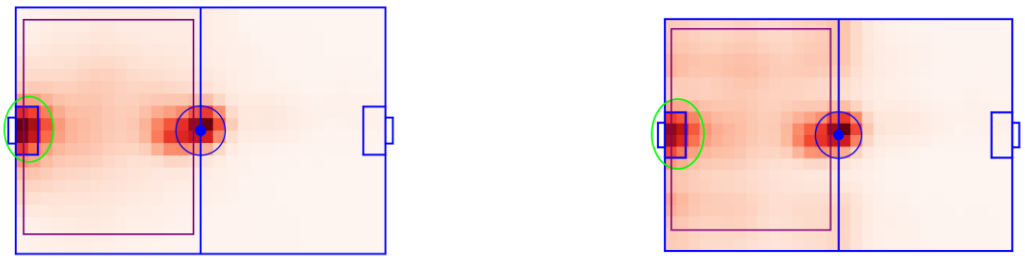


Figura 5.7: Mapa de calor da posição da bola e sua dispersão no campo durante o lote de 100 partidas do BahiaRT contra o magmaOffenburg sem e com a aprendizagem de *setplays*.

oponente, o magmaOffenburg marca menos gols por jogo.

Os resultados apresentados evidenciam clara influência da política de *setplays* aprendida pela abordagem apresentada nesta tese e na forma de jogar da equipe contra adversários de níveis diferentes. Na próxima seção, as considerações finais sobre os resultados são apresentados.

5.3 CONSIDERAÇÕES FINAIS

Este capítulo apresentou os experimentos e resultados da organização do dataset de demonstrações de *setplays*, deixando-o pronto para o treinamento do mecanismo de aprendizagem por reforço. A solução para a organização do dataset, apresentada nesta tese, mostrou-se eficaz, inclusive quanto à necessidade de divisão em dois níveis. Enquanto no primeiro nível a SF dos grupos não ultrapassava $0,5$, no nível 2 foi encontrado perfil de grupos com SF superior a $0,6$ evidenciando o bom resultado da abordagem produzida neste trabalho.

A organização dos datasets gerou 35 grupos de demonstrações, usados para treinar uma rede neural profunda com o algoritmo DQN. Neste capítulo, foi descrito todo o arranjo experimental utilizado para treinamento e testes da política de seleção de *setplays*. Os resultados evidenciaram que o conhecimento intuitivo coletado através das demonstrações pela estratégia de *crowdsourcing* podem influenciar a forma de atuação do Sistema Multiagentes (SMA). Os mapas de calor da posição da bola comprovaram a mudança no padrão de jogo da equipe.

O próximo capítulo apresenta as conclusões desta tese, o resumo das principais contribuições e uma análise das possibilidades de trabalhos futuros.

Capítulo

6

Neste capítulo, uma análise dos resultados obtidos é apresentada, bem como os prêmios, publicações e perspectivas de trabalhos futuros.

CONCLUSÕES

Os resultados apresentados nesta tese confirmam a hipótese inicial apresentada no início desta pesquisa:

É possível coletar de forma intuitiva, ao assistir um Sistema Multiagentes (SMA) em ação, um conjunto de demonstrações que permitem ao SMA aprender a usá-las para melhorar seu desempenho.

6.1 RESULTADOS OBTIDOS

Os resultados apresentados no capítulo 5, demonstram que foi possível coletar durante 4 meses, numa estratégia pública de *crowdsourcing*, um conjunto de demonstrações de jogadas coletivas para um time de futebol de robôs humanoides simulados da *RoboCup 3D Soccer Simulation* (3DSSIM). As demonstrações foram usadas numa estratégia de aprendizagem por reforço, utilizando redes neurais profundas, para aprender uma política de seleção de jogadas coletivas adequada às habilidades atuais do time.

Os experimentos utilizaram como referência o SMA desenvolvido pela equipe *Bahia Robotics Team* (BahiaRT) e comprovaram um ganho nas estatísticas de gols marcados e sofridos, além do avanço na posição média da bola durante as partidas contra três adversários selecionados na comunidade 3DSSIM a partir do histórico de confrontos dos mesmos com o BahiaRT. Os resultados foram comparados com a versão do time que não usa as demonstrações de *setplays* nem a política aprendida. Ao treinar a política de seleção de *setplays* atribuindo peso ao deslocamento horizontal da bola, foi possível aprender uma estratégia que seleciona os grupos de setplays que melhoram as chances do time levar a bola ao ataque, considerando as habilidades atuais dos jogadores na equipe. Mantendo a bola mais tempo no ataque, dificulta para as equipes adversárias marcarem gols, além de favorecer as oportunidades da equipe BahiaRT marcar seus gols. A

inclusão de *setplays* defensivos no dataset permitiu que a política aprendida utilizasse-os para criar dificuldades, através dos comportamentos de marcação, para equipes que trocam muitos passes como o magmaOffenburg, reduzindo a quantidade de gols sofridos contra esta equipe. Os *setplays* ofensivos que resultam em gols marcados durante o treinamento aumentam exponencialmente a recompensa favorecendo a escolha destes grupos na política aprendida, explicando o aumento na quantidade de gols marcados contra os adversários ITAndroids e WITS-FC.

Fica, portanto evidenciada a contribuição desta tese em apresentar um método de aprendizagem baseado em demonstrações coletadas de forma massiva, para permitir que um SMA possa aprender com este conhecimento humano não formalizado. A abordagem apresentada pode ser testada e validada em qualquer domínio de problema cuja solução seja implementada através de um SMA.

Vale ressaltar que, para a execução de um rede neural profunda para o aprendizado por reforço, os ambientes mais avançados usados pela comunidade científica como o OpenAI Gym e a biblioteca Stable Baselines 3 foram utilizados. Entretanto, não existia, publicamente disponível, nenhum ambiente Gym customizado para o simulador da 3DSSIM. Esta tese gerou, como um dos seus produtos, o BahiaRT Gym, ambiente customizado licenciado como software livre disponível publicamente, pronto para uso, não somente pelo BahiaRT mas por qualquer equipe da comunidade 3DSSIM. O BahiaRT Gym tem entre seus diferenciais um destaque para o fato de ter suporte para múltiplos agentes durante o treinamento. Então, foi possível utilizá-lo para treinar a seleção de *setplays* com o time BahiaRT completo em campo com seus onze jogadores e mais os oponentes. Estes últimos não interagem com o Gym mas exercem papel fundamental para compor o espaço de observação no treinamento.

Para viabilizar a estratégia de aprendizagem por reforço, esta tese produziu um método de organização das demonstrações para tratar a questão da equivalência semântica, definida neste trabalho. O organizador do conjunto de dados utilizou uma estratégia difusa de aprendizagem de máquina não supervisionada inovando no método (dividindo o processo em duas etapas para lidar com a complexidade das propriedades das instâncias) e na forma de cálculo de distâncias para lidar com dados não-escalares. A organização em grupos torna viável o projeto da aprendizagem por reforço, uma vez que num jogo de futebol de robôs o ciclo de simulação é de $20ms$, inviabilizando pelo agente a carga e seleção dos *setplays* a partir de um conjunto de dados massivo. A abordagem apresentada nesta tese dá a oportunidade do SMA realizar a seleção a partir de um grupo com uma quantidade reduzida de demonstrações que são apropriadas para a situação corrente no jogo. O dataset e o código do organizador são mais um produto e contribuição desta tese.

A estratégia de *crowdsourcing* foi materializada através de um kit de ferramentas denominado BahiaRT Setplays Collecting Toolkit. Neste trabalho, a implementação deste kit e disponibilização pública do mesmo para coleta das demonstrações foram realizadas. Para o desenvolvimento do kit, foi necessário gerar novas versões do visualizador de jogos oficial da 3DSSIM, o RoboViz, e da ferramenta de planejamento de estratégias *Strategy Planner* (SPlanner). Todo o trabalho foi reunido e documentado no kit de ferramentas apresentado a comunidade incluindo um vídeo tutorial detalhado de uso deste kit. Esta é mais uma contribuição deste trabalho para a comunidade que pode usar o kit para criar

documentações para uso em seus próprios experimentos ou submeter as demonstrações geradas, utilizando o formulário disponível no kit de ferramentas, visando a atualização do dataset em versões futuras.

6.2 PRÊMIOS E PUBLICAÇÕES

Esta tese gera um claro avanço no estado da arte, abrindo a possibilidade de vários prêmios e publicações. Uma possibilidade, inclusive já apresentada como proposta de trabalho para a comunidade da RoboCup Brasil durante o Desafio de Inovação Petrobras ocorrido durante a Competição Brasileira de Robótica 2021, é a aplicação da solução desta tese para um time de drones capaz de atuar em tarefas coletivas diversas numa plataforma de petróleo ou para solução de problemas ambientais. A proposta foi premiada com o segundo lugar neste desafio, demonstrando o interesse da comunidade nos resultados que esta tese apresenta.

Além disso, alguns resultados já geraram publicações em periódicos e conferências relevantes da área:

- Simões, Marco A. C. ; Nobre, Jadson ; Sousa, Gabriel ; Souza, Caroline ; Da Silva, Robson Marinho ; Campos, Jorge ; Souza, Josemar R. ; **Nogueira, T.** . Generating a dataset for learning setplays from demonstration. SN Applied Sciences, v. 3, n. 6, p. 608, jun. 2021.

- Simões, Marco A. C. ; Nobre, Jadson ; Sousa, Gabriel ; Souza, Caroline ; Da Silva, Robson Marinho ; Campos, Jorge ; Souza, Josemar R. ; **Nogueira, T.** . Strategy Planner: Enhancements to support better defense and pass strategies within an LfD approach. In: 2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). Ponta Delgada, Portugal: IEEE, 2020. p. 46–52.

- Simões, Marco A. C. ; Da Silva, Robson Marinho ; **Nogueira, T.** . A Dataset Schema for Cooperative Learning from Demonstration in Multi-robot Systems. Journal of Intelligent & Robotic Systems, v. 1, p. 1,2019.

- Simões, Marco A. C. ; **Nogueira, T** . Towards setplays learning in a multiagent robotic soccer team. In: 2018 Latin American Robotic Symposium, Brazilian Symposium on Robotics (SBR) and Workshop on Robotics in Education (WRE). [s.n.], 2018. p. 277– 282.

- Simões, Marco A. C. ; Mascarenhas, G. ; Fonseca, R. ; dos Santos, V. M. ; Mascarenhas, F. ; and **Nogueira, T.**. BahiaRT Setplays Collecting Toolkit and BahiaRT Gym. Software Impacts, 14:100401, 2022.

Como esperado, o proponente desta tese participou de várias competições com o SMA BahiaRT, alcançando resultado cada vez melhores, à medida que a pesquisa avançava, como pode ser observado pelas colocações obtidas nas competições brasileira e mundial mais conhecidas:

- Brazilian Robotics Competition / RoboCup Brazil Open

  - Campeão de 2017 a 2021

- RoboCup:
  - 2017: Sexto lugar
  - 2018: Quarto lugar
  - 2019: Quarto lugar
  - 2020: Devido à pandemia de COVID-19, a competição não ocorreu
  - 2021: Terceiro lugar no Desafio Técnico

6.3 TRABALHOS FUTUROS

A abordagem apresentada nesta tese pode ser estendida para qualquer domínio que envolva cooperação em SMA onde pessoas podem assistir os agentes em ação, possivelmente em uma ambiente simulado, e propor comportamentos coletivos coordenados que possam formar um conjunto de demonstrações para treinar uma política de seleção de comportamentos adequado para cada SMA.

Uma possibilidade futura inclui também a validação do efeito que as habilidades individuais de cada agente no SMA tem sobre o aprendizado das jogadas coletivas. O time BahiaRT não tem todas as suas habilidades no estado da arte da comunidade 3DSSIM. Existem projetos em andamento para aprimorar habilidades básicas como caminhada, chute, habilidades de goleiro, etc. Quando estes projetos gerarem resultados, pode-se utilizar a mesma base de demonstrações desta tese para treinar novamente a política de seleção de *setplays* e comparar os resultados com os apresentados nesta tese. Da mesma forma, outras equipe podem usar o dataset gerado para treinar suas próprias políticas de seleção de *setplays* e comparar os resultados com os deste trabalho e com o próprio desempenho anterior de seus times. O entendimento do limite em que a contribuição das habilidades individuais para desempenho global do SMA influencia no comportamento coletivo é essencial para o projeto e aplicação dos SMA em diversos domínios de problema.

O uso a modelagem de oponentes como um dos critérios para treinamento da política de seleção de *setplays* abre uma possibilidade interessante de trabalhos futuros. Esta abordagem permitirá que a política adapte-se à forma de jogar time adversários, podendo adaptar-se dinamicamente às condições ambientais do problema.

A própria solução aqui apresentada pode ainda ser tratada em diversos experimentos em trabalhos em nível de graduação, mestrado ou doutorado. A organização do dataset pode ser avaliada e aperfeiçoada utilizando outros Índices de Validação de Clusters (IVCs), além de ter as normas de distância melhoradas para refletir ainda mais o perfil de dados desejado. Variações nos hiper-parâmetros do algoritmo *Deep Q Network* (DQN) podem ser testadas e validadas. O uso de outros algoritmos de aprendizagem profunda no lugar do DQN também pode ser alvo de investigações futuras. Outra proposta de investigação é a modificação do framework de *setplays* para substituir o mecanismo de Raciocínio Baseado em Casos (RBC) por uma rede neural profunda que já selecione diretamente o *setplay* desejado, ao invés de selecionar o grupo. As questões de desempenho e requisitos de tempo da aplicação precisam ser avaliadas num cenário destes. O BahiaRT

Gym, publicado como software livre, pode ter novas funcionalidades e ajustes desenvolvidos pela comunidade, tornando-o alvo potencial também para trabalhos futuros.

Em resumo, a gama de possibilidades de pesquisa aberta por esta tese é extensa, caracterizando claramente as muitas contribuições científicas e acadêmicas da mesma. Com os resultados aqui apresentados, amplia-se a fronteira do conhecimento nas áreas de Sistema Multiagentes (SMA), aprendizagem de máquina e robótica.

REFERÊNCIAS BIBLIOGRÁFICAS

ABREU, M. et al. Learning low level skills from scratch for humanoid robot soccer using deep reinforcement learning. In: *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. [S.l.: s.n.], 2019. p. 1–8.

ABREU, M.; REIS, L. P.; LAU, N. Learning to run faster in a humanoid robot soccer environment through reinforcement learning. In: *RoboCup 2019: Robot World Cup XXIII*. Sydney: RoboCup Federation, 2019. v. 1. Disponível em: ⟨https://2019.robocup.org/downloads/program/AbreuEtAl2019.pdf⟩.

ABREU, M. et al. 6D Localization and Kicking for Humanoid Robotic Soccer. *Journal of Intelligent & Robotic Systems*, v. 102, n. 2, p. 30, maio 2021. ISSN 1573-0409. Disponível em: ⟨https://doi.org/10.1007/s10846-021-01385-3⟩.

ALMEIDA, F. et al. An automatic approach to extract goal plans from soccer simulated matches. *Soft Computing*, v. 17, n. 5, p. 835–848, 2013. ISSN 1433-7479. Disponível em: ⟨https://doi.org/10.1007/s00500-012-0952-z⟩.

ANTãO, L. et al. Learning to Play Precision Ball Sports from scratch: a Deep Reinforcement Learning Approach. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2020. p. 1–8. ISSN: 2161-4407.

ARGALL, B. D. et al. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, v. 57, n. 5, p. 469–483, maio 2009. ISSN 09218890. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0921889008001772⟩.

BEZDEK, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 1981. ISBN 978-0-306-40671-3.

BIANCHI, R. A. et al. Heuristically accelerated reinforcement learning by means of case-based reasoning and transfer learning. *Journal of Intelligent & Robotic Systems*, v. 91, n. 2, p. 301–312, 2018. ISSN 1573-0409. Disponível em: ⟨https://doi.org/10.1007/s10846-017-0731-2⟩.

BOEDECKER, J.; ASADA, M. SimSpark – Concepts and Application in the RoboCup 3D Soccer Simulation League. *Autonomous Robots*, v. 174, p. 181–188, 2008.

BROCKMAN, G. et al. *OpenAI Gym*. arXiv.org, 2016. _eprint: arXiv:1606.01540. Disponível em: ⟨http://arxiv.org/abs/1606.01540⟩.

CAMPELLO, R.; HRUSCHKA, E. A fuzzy extension of the silhouette width criterion for cluster analysis. *Fuzzy Sets and Systems*, v. 157, n. 21, p. 2858–2875, 2006. ISSN 01650114. Disponível em: ⟨https://www.scopus.com/inward/record.uri?eid=2-s2.0-33749142135&doi=10.1016\%2fj.fss.2006.07.006&partnerID=40&md5=383e6afe20ad96ea4b9e8bfda1f14ebe⟩.

COSTA, A. C. P. L. da; BITTENCOURT, G. UFSC-team: A Cognitive Multi-agent Approach to the RoboCup'98 Simulator League. In: ASADA, M.; KITANO, H. (Ed.). *RoboCup-98: Robot Soccer World Cup II*. [S.l.]: Springer Berlin Heidelberg, 1999. (Lecture Notes in Computer Science), p. 371–376. ISBN 978-3-540-48422-6.

COSTA, A. L. da; BITTENCOURT, G. From a Concurrent Architecture to a Concurrent Autonomous Agents Architecture. In: VELOSO, M.; PAGELLO, E.; KITANO, H. (Ed.). *RoboCup-99: Robot Soccer World Cup III*. [S.l.]: Springer Berlin Heidelberg, 2000. (Lecture Notes in Computer Science), p. 274–285. ISBN 978-3-540-45327-7.

CRAVO, J. et al. Strategy planner: Graphical definition of soccer set-plays. *Data & Knowledge Engineering*, v. 94, p. 110–131, nov. 2014. ISSN 0169-023X. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S0169023X14000950⟩.

D'AMBROSIO, D. B.; STANLEY, K. O. Scalable multiagent learning through indirect encoding of policy geometry. *Evolutionary Intelligence*, v. 6, n. 1, p. 1–26, jun. 2013. ISSN 1864-5917. Disponível em: ⟨https://doi.org/10.1007/s12065-012-0086-3⟩.

EUSTáQUIO, F. et al. On Fuzzy Cluster Validity Indexes for High Dimensional Feature Space. In: KACPRZYK, J. et al. (Ed.). *Advances in Fuzzy Logic and Technology 2017*. [S.l.]: Springer International Publishing, 2018. (Advances in Intelligent Systems and Computing), p. 12–23. ISBN 978-3-319-66824-6.

EUSTáQUIO, F.; NOGUEIRA, T. On Monotonic Tendency of Some Fuzzy Cluster Validity Indices for High-Dimensional Data. In: *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*. [S.l.: s.n.], 2018. p. 558–563.

FABRO, J. A.; REIS, L. P.; LAU, N. Using Reinforcement Learning Techniques to Select the Best Action in Setplays with Multiple Possibilities in Robocup Soccer Simulation Teams. In: *2014 Joint Conference on Robotics: SBR-LARS Robotics Symposium and Robocontrol*. Sao Carlos, Sao Paulo, Brazil: IEEE, 2014. p. 85–90. ISBN 978-1-4799-6711-7 978-1-4799-6710-0. Disponível em: ⟨http://ieeexplore.ieee.org/document/7024261/⟩.

FREELAN, D. et al. Towards Rapid Multi-robot Learning from Demonstration at the RoboCup Competition. In: *RoboCup 2014: Robot World Cup XVIII*. Springer, Cham, 2014. (Lecture Notes in Computer Science), p. 369–382. ISBN 978-3-319-18614-6 978-3-319-18615-3. Disponível em: ⟨https://link.springer.com/chapter/10.1007/978-3-319-18615-3_30⟩.

HIEU, D. V.; MEESAD, P. A cell-MST-based method for big dataset clustering on limited memory computers. In: *2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE)*. [S.l.: s.n.], 2015. p. 632–637.

HIEU, D. V.; MEESAD, P. Fast K-Means Clustering for Very Large Datasets Based on MapReduce Combined with a New Cutting Method. In: NGUYEN, V.-H.; LE, A.-C.; HUYNH, V.-N. (Ed.). *Knowledge and Systems Engineering*. Cham: Springer International Publishing, 2015. v. 326, p. 287–298. ISBN 978-3-319-11679-2 978-3-319-11680-8. Disponível em: ⟨http://link.springer.com/10.1007/978-3-319-11680-8_23⟩.

Júnior, Orivaldo Vieira Santana; da Costa, Augusto Loureiro. MecaTeam 2006: Um Sistema Multiagente Reativo para o futebol de robôs simulado. In: *Anais do XXVI Congresso da Sociedade Brasileira de Computação*. [s.n.], 2006. p. 146–152. Disponível em: ⟨https://pessoal.ect.ufrn.br/~orivaldo/publicacoes/SBC-ENRI_mecateam2006.pdf⟩.

KALYANAKRISHNAN, S. et al. Three humanoid soccer platforms: Comparison and synthesis. In: BALTES, J. et al. (Ed.). *RoboCup 2009: Robot soccer world cup XIII*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 140–152. ISBN 978-3-642-11876-0.

KASAEI, M. et al. Robust biped locomotion using deep reinforcement learning on top of an analytical control approach. *Robotics and Autonomous Systems*, v. 146, p. 103900, dez. 2021. ISSN 09218890. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0921889021001858⟩.

KITANO, H. et al. RoboCup: A challenge problem for AI and robotics. In: KITANO, H. (Ed.). *RoboCup-97: Robot soccer world cup I*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998. p. 1–19. ISBN 978-3-540-69789-3.

LIEMHETCHARAT, S.; VELOSO, M. Allocating training instances to learning agents for team formation. *Autonomous Agents and Multi-Agent Systems*, v. 31, n. 4, p. 905–940, jul. 2017. ISSN 1573-7454. Disponível em: ⟨https://doi.org/10.1007/s10458-016-9355-3⟩.

LIN, L.-J. *Reinforcement learning for robots using neural networks*. [S.l.]: Carnegie Mellon University, 1992.

MELO, L. C.; MELO, D. C.; MAXIMO, M. R. O. A. Learning Humanoid Robot Running Motions with Symmetry Incentive through Proximal Policy Optimization. *Journal of Intelligent & Robotic Systems*, v. 102, n. 3, p. 54, jun. 2021. ISSN 1573-0409. Disponível em: ⟨https://doi.org/10.1007/s10846-021-01355-9⟩.

MICALIZIO, R.; TORTA, G. Explaining interdependent action delays in multiagent plans execution. *Autonomous Agents and Multi-Agent Systems*, v. 30, n. 4, p. 601–639, jul. 2016. ISSN 1573-7454. Disponível em: ⟨https://doi.org/10.1007/s10458-015-9298-0⟩.

MITCHELL, T. M. *Machine learning*. 1st. ed. New York: McGraw-Hill, 1997. (McGraw-Hill series in Computer Science). ISBN 978-0-07-115467-3 978-0-07-042807-2.

MIYASHITA, Y.; SUGAWARA, T. Analysis of coordinated behavior structures with multi-agent deep reinforcement learning. *Applied Intelligence*, 2020. ISSN 1573-7497. Disponível em: ⟨https://doi.org/10.1007/s10489-020-01832-y⟩.

MNIH, V. et al. Human-level control through deep reinforcement learning. *Nature*, v. 518, n. 7540, p. 529–533, fev. 2015. ISSN 0028-0836, 1476-4687. Disponível em: ⟨http://www.nature.com/articles/nature14236⟩.

MORADI, M.; ARDESTANI, M. A.; MORADI, M. Learning decision making for Soccer Robots: A crowdsourcing-based approach. In: *2016 Artificial Intelligence and Robotics (IRANOPEN)*. [S.l.: s.n.], 2016. p. 25–29.

MOTA, L. et al. Collaborative Behavior in Soccer: The Setplay Free Software Framework. In: HUTCHISON, D. et al. (Ed.). *Applied Cryptography and Network Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015. v. 7341, p. 709–716. ISBN 978-3-642-31283-0 978-3-642-31284-7. Disponível em: ⟨http://link.springer.com/10.1007/978-3-319-18615-3_58⟩.

MOTA, L.; LAU, N.; REIS, L. P. Co-ordination in RoboCup's 2D simulation league: Setplays as flexible, multi-robot plans. In: *2010 IEEE Conference on Robotics, Automation and Mechatronics*. [S.l.: s.n.], 2010. p. 362–367.

MOTA, L.; REIS, L. P.; LAU, N. Multi-robot coordination using Setplays in the middle-size and simulation leagues. *Mechatronics*, v. 21, n. 2, p. 434–444, mar. 2011. ISSN 09574158. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0957415810000851⟩.

NAYAK, J.; NAIK, B.; BEHERA, H. S. Fuzzy C-Means (FCM) Clustering Algorithm: A Decade Review from 2000 to 2014. In: JAIN, L. C. et al. (Ed.). *Computational Intelligence in Data Mining - Volume 2*. [S.l.]: Springer India, 2015. p. 133–149. ISBN 978-81-322-2208-8.

OCANA, J. M. C. et al. Cooperative Multi-Agent Deep ReinforcementLearning in a 2 Versus 2 Free-Kick Task. In: . Sydney: RoboCup Federation, 2019. v. 1. Disponível em: ⟨https://2019.robocup.org/downloads/program/OcanaEtAl2019.pdf⟩.

PANELLA, A.; GMYTRASIEWICZ, P. Interactive POMDPs with finite-state models of other agents. *Autonomous Agents and Multi-Agent Systems*, v. 31, n. 4, p. 861–904, jul. 2017. ISSN 1573-7454. Disponível em: ⟨https://doi.org/10.1007/s10458-016-9359-z⟩.

PRUDENCIO, R. F.; MAXIMO, M. R. O. A.; COLOMBINI, E. L. A Survey on Offline Reinforcement Learning: Taxonomy, Review, and Open Problems. *arXiv:2203.01387 [cs, stat]*, mar. 2022. ArXiv: 2203.01387. Disponível em: ⟨http://arxiv.org/abs/2203.01387⟩.

RAFFIN, A. et al. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 2021. Publisher: MIT Press.

RAPOPORT, A.; CHAMMAH, A. M.; ORWANT, C. J. *Prisoner's Dilemma: A Study in Conflict and Cooperation*. [S.l.]: University of Michigan Press, 1965. Google-Books-ID: yPtNnKjXaj4C. ISBN 978-0-472-06165-5.

REIS, L. P.; LAU, N.; OLIVEIRA, E. C. Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents. In: HANNEBAUER, M.; WENDLER, J.; PAGELLO, E. (Ed.). *Balancing Reactivity and Social Deliberation in Multi-Agent Systems*. [S.l.]: Springer Berlin Heidelberg, 2001. (Lecture Notes in Computer Science), p. 175–197. ISBN 978-3-540-44568-5.

REIS, L. P. et al. Playmaker: Graphical definition of formations and setplays. In: *5th Iberian Conference on Information Systems and Technologies*. [S.l.: s.n.], 2010. p. 1–6.

ROS, R. et al. A case-based approach for coordinated action selection in robot soccer. *Artificial Intelligence*, v. 173, n. 9-10, p. 1014–1039, jun. 2009. ISSN 00043702. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0004370209000319⟩.

RUSSELL, S. J.; NORVIG, P. *Artificial intelligence: a modern approach*. Fourth edition. Hoboken, NJ: Pearson, 2021. (Pearson Series in Artificial Intelligence). ISBN 978-0-13-461099-3.

RöFER, T. et al. B-Human team report and code release 2017 (2017). *Only available online: http://www. b-human. de/downloads/publications/2017/coderelease2017. pdf*.

SARDAR, T. H.; ANSARI, Z.; KHATUN, A. An evaluation of Hadoop cluster efficiency in document clustering using parallel K-means. In: *2017 IEEE International Conference on Circuits and Systems (ICCS)*. [S.l.: s.n.], 2017. p. 17–20.

SHAO, W.; SHI, X.; YU, P. S. Clustering on Multiple Incomplete Datasets via Collective Kernel Learning. In: *2013 IEEE 13th International Conference on Data Mining*. Dallas, TX, USA: IEEE, 2013. p. 1181–1186. ISBN 978-0-7695-5108-1. Disponível em: ⟨http://ieeexplore.ieee.org/document/6729618/⟩.

SHI, H. et al. An Adaptive Strategy Selection Method With Reinforcement Learning for Robotic Soccer Games. *IEEE Access*, v. 6, p. 8376–8386, 2018. ISSN 2169-3536.

SILVA, F. L.; COSTA, A. H. R.; STONE, P. Distributional Reinforcement Learning Applied to Robot Soccer Simulation. In: *Proc. of the 2019 Adaptive and Learning Agents*. Montreal, Canada: [s.n.], 2019. v. 1, p. 1–5.

SIMõES, M. A. C. et al. Strategy Planner: Enhancements to support better defense and pass strategies within an LfD approach. In: *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. Ponta Delgada, Portugal: IEEE, 2020. p. 46–52. ISBN 978-1-72817-078-7. Disponível em: ⟨https://ieeexplore.ieee.org/document/9096188/⟩.

SIMõES, M. A. C. et al. Generating a dataset for learning setplays from demonstration. *SN Applied Sciences*, v. 3, n. 6, p. 608, jun. 2021. ISSN 2523-3963, 2523-3971. Disponível em: ⟨https://link.springer.com/10.1007/s42452-021-04571-y⟩.

SIMõES, M. A. C.; NOGUEIRA, T. Towards setplays learning in a multiagent robotic soccer team. In: *2018 latin american robotic symposium, 2018 brazilian symposium on robotics (SBR) and 2018 workshop on robotics in education (WRE)*. [s.n.], 2018. p. 277–282. ISBN 978-1-5386-7761-2. Disponível em: ⟨https://doi.org/10.1109/LARS/SBR/WRE.2018.00058⟩.

SIMõES, M. A. C.; SILVA, R. M. da; NOGUEIRA, T. A Dataset Schema for Cooperative Learning from Demonstration in Multi-robot Systems. *Journal of Intelligent & Robotic Systems*, v. 99, n. 3-4, p. 589–608, 2020. ISSN 0921-0296, 1573-0409. Publisher: Springer Science and Business Media LLC. Disponível em: ⟨http://link.springer.com/10.1007/s10846-019-01123-w⟩.

SPITZNAGEL, M.; WEILER, D.; DORER, K. Deep Reinforcement Multi-Directional Kick-Learning of a Simulated Robot with Toes. In: *2021 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. [S.l.: s.n.], 2021. p. 104–110.

STONE, P.; VELOSO, M. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*, v. 110, n. 2, p. 241–273, jun. 1999. ISSN 00043702. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0004370299000259⟩.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: an introduction*. Second edition. Cambridge, Massachusetts: The MIT Press, 2018. (Adaptive computation and machine learning series). ISBN 978-0-262-03924-6.

TEIXEIRA, H. et al. Humanoid Robot Kick in Motion Ability for Playing Robotic Soccer. In: *2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. Ponta Delgada, Portugal: IEEE, 2020. p. 34–39. ISBN 978-1-72817-078-7. Disponível em: ⟨https://ieeexplore.ieee.org/document/9096073/⟩.

WANGENHEIM, C. G. v.; WANGENHEIM, A. v. *Raciocínio baseado em casos*. Barueri: Manole, 2003. OCLC: 69935690. ISBN 978-85-204-1459-0.

WEISS, G. (Ed.). *Multiagent systems: a modern approach to distributed artificial intelligence*. Cambridge, Mass: MIT Press, 1999. ISBN 978-0-262-23203-6.

WOOLDRIDGE, M. *An Introduction to Multiagent Systems*. 2. ed. Chichester, UK: Wiley, 2009. ISBN 978-0-470-51946-2.

WóJCIK, M. R.; ZDUNEK, R.; ANTOńCZAK, A. J. Unsupervised verification of laser-induced breakdown spectroscopy dataset clustering. *Spectrochimica Acta Part B: Atomic Spectroscopy*, v. 126, p. 84–92, dez. 2016. ISSN 05848547. Disponível em: ⟨https://linkinghub.elsevier.com/retrieve/pii/S0584854716302774⟩.

XU, X.; LI, D.; ZHONG, C. Fuzzy clustering based on re-classification of border data for incomplete dataset. In: *2017 36th Chinese Control Conference (CCC)*. [S.l.: s.n.], 2017. p. 10777–10782.

YANG, Y. S. et al. Hybrid Genetic Clustering by Using FCM and Geodesic Distance for Complex Distributed Data. *Applied Mechanics and Materials; Zurich*, v. 263-266, dez. 2012. ISSN 16609336. Disponível em: ⟨https://search.proquest.com/docview/1442793660/abstract/A25A7D33D8ED4F71PQ/1⟩.

YU, C. et al. Emotional Multiagent Reinforcement Learning in Spatial Social Dilemmas. *IEEE Transactions on Neural Networks and Learning Systems*, v. 26, n. 12, p. 3083–3096, 2015. ISSN 2162-237X.

YU, C. et al. Multiagent Learning of Coordination in Loosely Coupled Multiagent Systems. *IEEE Transactions on Cybernetics*, v. 45, n. 12, p. 2853–2867, 2015. ISSN 2168-2267.

ZHANG, C.; SINHA, A.; TAMBE, M. Keeping Pace with Criminals: Designing Patrol Allocation Against Adaptive Opportunistic Criminals. In: *roceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. Istanbul, Turkey: [s.n.], 2015. p. 1351–1359. ISBN 978-1-4503-3413-6.

ZHANG, C.; TAMBE, M. Modeling, Learning and Defending against Opportunistic Criminals in Urban Areas (Doctoral Consortium). In: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. Istambul, Turkey: [s.n.], 2015. p. 1971–1972. ISBN 978-1-4503-3413-6.

ZHANG, S. et al. Multirobot Symbolic Planning under Temporal Uncertainty. In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. São Paulo, Brazil: [s.n.], 2017. p. 501–510.

ZHOU, J.; PURVIS, M.; MUHAMMAD, Y. A Combined Modelling Approach for Multi-Agent Collaborative Planning in Global Supply Chains. In: *2015 8th International Symposium on Computational Intelligence and Design (ISCID)*. [S.l.: s.n.], 2015. v. 1, p. 592–597.

ZIANI, D. Feature selection on probabilistic symbolic objects. *Frontiers of Computer Science*, v. 8, n. 6, p. 933–947, 2014. ISSN 2095-2236. Disponível em: ⟨https://doi.org/10.1007/s11704-014-3359-4⟩.