Universidade Federal da Bahia
Escola Politécnica / Instituto de Matemática
Programa de Pós-graduação em Mecatrônica

UM MÉTODO PARA OBTENÇÃO DO MODELO COMPORTAMENTAL DO TIME OPONENTE EM UMA PARTIDA DE FUTEBOL ENTRE EQUIPES DE ROBÔS AUTÔNOMOS

Marcelo Santos Linder

Salvador
2008

Universidade Federal da Bahia

Marcelo Santos Linder

UM MÉTODO PARA OBTENÇÃO DO MODELO COMPORTAMENTAL DO TIME OPONENTE EM UMA PARTIDA DE FUTEBOL ENTRE EQUIPES DE ROBÔS AUTÔNOMOS

Dissertação apresentada ao Programa de Pós-graduação em Mecatrônica, programa conjunto entre o Departamento de Engenharia Mecânica e o Departamento de Ciência da Computação, da Universidade Federal da Bahia como requisito parcial para obtenção do grau de Mestre em Mecatrônica.

Orientador: Prof. Dr. Augusto Loureiro da Costa.

Salvador
2008

L744m Linder, Marcelo Santos

Um método para obtenção do modelo comportamental do time oponente em uma partida de futebol entre equipes de robôs autônomos / Marcelo Santos Linder. – 2008.

166 f. : il.

Dissertação (Mestrado) – Universidade Federal da Bahia, Escola Politécnica / Instituto de Matemática, 2008.

Orientador: Prof. Dr. Augusto Loureiro da Costa.

1. Robôs. 2. Modelagem Comportamental. 3. Sistemas Multiagentes I. Costa, Augusto Loureiro da

CDD 629.892
CDU 681.3

Universidade Federal da Bahia

Marcelo Santos Linder

UM MÉTODO PARA OBTENÇÃO DO MODELO COMPORTAMENTAL DO TIME OPONENTE EM UMA PARTIDA DE FUTEBOL ENTRE EQUIPES DE ROBÔS AUTÔNOMOS

Banca Examinadora

Augusto Loureiro da Costa (Orientador)

Débora Abdalla Santos (Examinador)

Evandro de Barros Costa (Examinador)

Leizer Schnitman (Examinador)

Salvador
2008

AGRADECIMENTOS

Aos meus pais que me geraram, me educaram através do exemplo e, sempre confiantes, estimularam meu potencial. A minha mãe, Rita Maria Santos Moraes, que sempre foi um consolo para minhas aflições e um ouvido para minhas lamúrias. A meu pai, Edison Linder de Moraes, pelo exemplo e pelas constantes cobranças de quando eu iria acabar o mestrado.

A Michel, meu grande irmão, que sempre foi um companheiro para qualquer adversidade.

A meu filho Gabrielzinho cujo simples fato de existir enche minha vida de todos os mais puros sentimentos e que é o responsável por aqueles pensamentos alegres que surgem em minha mente nos momentos de cansaço e frustração e são capazes de transformar minha expressão em um lindo sorriso motivando-me a superar obstáculos.

A Marília minha ex-esposa que em diversos momentos foi um apoio para meus fraquejos.

A Catarina minha companheira, adjetivo este muito modesto para representar o que ela significa para mim, uma vez que, ela constitui todos os meus ideais de uma mulher para envelhecer ao lado, compartilhando todos os momentos de minha existência.

Aos meus grandes amigos Cássio Carvalho, Jean Orengo e Gustavo Sabin, companheiros de muitas empreitadas no período da graduação, situações estas que contribuíram para o meu crescimento profissional.

Ao Programa de Pós-Graduação em Mecatrônica da Universidade Federal da Bahia pela confiança em mim depositada e, em especial, agradeço ao meu orientador Augusto Loureiro da Costa pelos valiosos conselhos e contribuições no desenvolvimento do trabalho.

A CAPES pela bolsa que me foi concedida possibilitando minha dedicação integral no desenvolvimento do trabalho.

A Universidade Federal do Vale do São Francisco pelo afastamento a mim concedido, fato determinante para a finalização de meu mestrado.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

IAD	Inteligência Artificial Distribuída
IA	Inteligência Artificial
LSR	Liga Simulada 2d da RoboCup
MDA	Mensagem de Descrição do Ambiente
MOMCoTO	Método para Obtenção do Modelo Comportamental do Time Oponente
OMTU	Operações Militares em Terreno Urbano
RdP	Rede de Petri
SMA	Sistema Multiagente

RESUMO

Sistemas Multiagentes estão cada vez mais presentes em diversas aplicações como em missões espaciais com sistemas multirobôs, em simulações de confrontos militares, tráfego de veículos e pedestres. A identificação das interações entre agentes em um ambiente é uma atividade complexa, assim como analisar o comportamento inteligente que emerge a partir das interações entre os agentes. Contudo, obter um modelo comportamental de um Sistema Multiagente é necessário para que a partir deste se possa analisar o sistema. Partindo desta premissa, o presente trabalho, propôs, implementou e validou um método para obtenção do modelo comportamental de um Sistema Multiagente. Mais especificamente, para obtenção do modelo comportamental do time oponente, em uma partida de futebol entre equipes de robôs autônomos. Tal método baseia-se em através de uma análise dos dados obtidos por intermédio de visões globais do ambiente, classificar o estado do ambiente, identificando ações dos agentes sobre o ambiente e com base nestas informações gerar, simultaneamente, com o andamento das transformações do ambiente, um modelo comportamental de um grupo de agentes imerso no ambiente. Foi escolhido como formalismo para representação do modelo comportamental uma Rede de Petri, a qual descreve o comportamento do time adversário.

Palavras-chave: Sistemas Multiagentes, Modelagem Comportamental, Redes de Petri e Futebol de Robôs.

ABSTRACT

The Multi-Agent System Applications have been increased a lot in such fields like Multi-Robot System for Industrial, Environment Monitoring or Space Missions, Simulation of Military Operations; Traffic Control, etc. The agent interaction identification is a very complex process as such as the intelligent behavior analyses emerged from the interaction of a set of agents in a given environment. Otherwise a multi-agent system behavior model is very important to ensure the right performing of the system or tho analise the multi-agent system behavior. The present work presents a methodology for multi-agent system behavior modeling. Both the modeling proposal and its implementation are presented in this work followed some experimental results. The Robot Soccer simulation domain was taken as experimental environment and the team behavior was chosen as case study for modeling. The methodology is based on analyses from the global see messages sent from the RoboCup Soccer Server Simulation to the coach agent at each Robot Soccer Server Simulator simulation cycle. This messages contains a framework who describes the environment from a godview point of view. Once each new received framework the environment state is classified, and the agent actions identified, and a Petri Network is dynamically built in during the match. The Petri Net stores the multi-agent system behavior model.

Keywords: Multi-Agent System, Behavioral Modeling, Petri Networks and Robot Soccer.

1. INTRODUÇÃO

A Inteligência Artificial Distribuída (IAD) é uma das áreas da Inteligência Artificial (IA) que mais se desenvolveu nos últimos anos e apresenta um enorme potencial de aplicação (CHENY ET AL, 2002). Segundo (WEISS, 1999), esta área envolve o estudo, construção e aplicação de sistemas multiagentes, onde agentes inteligentes (entidades lógicas ou físicas), interagem para realizar um conjunto de objetivos ou tarefas, como ocorrem em missões espaciais com sistemas multirobôs (DORAIS ET AL, 1999), em simulações de confrontos militares, tráfego de pedestres (SILVA, 2003), etc.

A implementação de sistemas multiagentes pressupõe a escolha de uma arquitetura para os agentes e a eleição do protocolo de interação que permitirá aos agentes formar uma sociedade de agentes, caracterizando um exemplo de Sistema Multiagente (SMA). Diferentes arquiteturas e protocolos de interação já foram propostos, o que permite a implementação de um SMA segundo diversas metodologias. Uma dificuldade histórica inerente à área da IA e por conseqüência na IAD, sempre foi a comparação entre as diferentes técnicas utilizadas para implementação de soluções que utilizem técnicas de IA ou IAD.

Uma forma comum de incentivo à pesquisa em IA, assim como em IAD, é a proposta de um problema padrão. Problema que permite a utilização de diferentes técnicas de IA e que possibilita, principalmente, a comparação entre diferentes soluções. No campo da IA o jogo de xadrez é um exemplo de problema padrão utilizado nas décadas de 70 e 80 para comparar soluções.

Um problema bastante interessante, que vem atraindo interesse de diversos grupos de pesquisa em robótica e IAD, é a realização de partidas de futebol entre robôs autônomos. Esta proposta surgiu da iniciativa de um grupo de pesquisadores em IA e robótica móvel, que em 1997 resultou na Primeira Copa Mundial de Futebol de Robôs, a RoboCup (ROBOCUP, 2007), em Nagoya, Japão.

Reconhecendo a potencialidade deste problema, um grupo de pesquisa da Universidade Federal da Bahia (UFBA) implementou dois projetos de pesquisa para

se dedicar a este tema, o AxeBot (AXEBOT, 2007) e o MecaTeam (MECATEAM, 2007), que se utilizam do domínio do futebol de robôs como laboratório para o ensino e pesquisa da mecatrônica e automação industrial. Estes projetos promovem a integração de várias tecnologias e áreas de pesquisa distintas, como: agentes inteligentes, sistemas multiagentes, estratégias de aquisição de conhecimento, sistemas de tempo real, sistemas distribuídos, reconhecimento de padrões, integração de sensores, controle de processos, sistemas embarcados, entre outras.

Em virtude das características do problema e dos desafios envolvidos numa partida de futebol, o futebol de robôs tornou-se um importante domínio, principalmente no que diz respeito à pesquisa na área de sistemas multiagentes. Em uma analise inicial pode-se dividir os trabalhos que envolvem times de agentes autônomos em dois grandes grupos:

- os que dedicam-se ao estudo dos aspectos que levam os agentes a cooperar na realização de uma tarefa coletiva;
- e os que se propõem a analisar o comportamento inteligente que emerge a partir das interações entre os agentes.

1.1. Modelagem Comportamental de Sistemas Multiagentes

Analisar um SMA identificando o comportamento inteligente que emerge a partir das interações entre os agentes pode ser caracterizado pela obtenção do modelo comportamental do sistema em questão. Tarefa esta que representa um grande desafio, segundo (NAIR; TAMBE, 2002), sistemas multiagentes com centenas ou mesmo uma dezena de agentes, têm um grande número de interações, altamente dinâmicas e complexas, tornando difícil a análise comportamental desses sistemas por humanos.

Obter o modelo comportamental de um SMA pode ser considerado, em determinada abordagem, um passo inicial no processo de estudo dos aspectos que levam os agentes a cooperar na realização de uma tarefa coletiva.

Como foi mencionado anteriormente, podem ser utilizados SMA´s para representar diversos sistemas como, por exemplo, para retratar confrontos militares, onde podemos nos valer da modelagem comportamental deste sistema para identificarmos o comportamento defensivo e/ou ofensivo do exército oponente, no qual podem ser consideradas como entradas para a modelagem comportamental informações oriundas de sensores de movimento instalados no campo de combate.

Outro exemplo de aplicação da modelagem comportamental de SMA seria na retratação de um sistema de tráfego de veículos em uma cidade, onde os automóveis seriam os agentes autônomos e através da modelagem comportamental do sistema poderiam ser identificadas as vias mais congestionadas, assim como as subutilizadas, podendo também serem identificados os cruzamentos onde ocorreram mais acidentes e etc.

Sendo mais especifico, particularizando o domínio de aplicação para o universo do futebol entre robôs autônomos, obter o modelo comportamental de um SMA que representa um ambiente de competição entre equipes, torna-se uma grande vantagem competitiva, ao considerarmos que um time em posse do modelo do SMA, no qual está inserido, passa, por exemplo, a ter a capacidade de antecipar as jogadas (ações) de seu adversário. Pois, com base no histórico comportamental do seu oponente, o time pode: identificar os agentes responsáveis pelas finalizações das jogadas, identificar a região do campo onde estas finalizações ocorrem, caracterizar as jogadas ensaiadas do time adversário e etc.. Além da possibilidade de absorção das características positivas identificadas no comportamento do time oponente.

1.2. Objetivos e Justificativas

Justificado pela necessidade da existência de ferramentas que auxiliem ou até mesmo façam a análise do comportamento de um SMA gerando um modelo comportamental do sistema em questão, este trabalho apresenta a concepção, projeto e implementação de um método para obtenção do modelo comportamental de um SMA.

De uma forma mais específica, este trabalho teve como objetivo, propor um método capaz de, com base em uma análise dos dados obtidos através de visões globais do ambiente, realizar as seguintes funcionalidades:

- classificar o estado do ambiente;
- identificar ações dos agentes sobre o ambiente;
- com base nas ações identificadas, caracterizar comportamentos e interações entre os agentes;
- e partindo das informações adquiridas, gerar, simultaneamente com o andamento das transformações do ambiente, um modelo comportamental de um grupo de agentes imerso no ambiente.

O ambiente em questão será o futebol entre robôs, onde o SMA será composto pelos times envolvidos no confronto e o modelo comportamental a ser obtido refletirá o comportamento do time oponente.

Esta dissertação está estruturada da seguinte forma: no capítulo 2 será feita a descrição do problema a ser solucionado; sendo no capítulo 3 discutidos trabalhos correlatos ao tema; o capítulo 4, abordará o método proposto para obtenção do modelo comportamental do time oponente; o capítulo 5 trata da implementação do método proposto apresentando um conjunto de estudos de caso; finalmente, o capítulo 6 apresenta conclusões e discorre sobre perspectivas para trabalhos futuros.

2. DESCRIÇÃO DO PROBLEMA

Com base no que foi exposto, o futebol entre equipes de robôs foi escolhido como domínio de aplicação para o método proposto neste trabalho já que este caracteriza adequadamente um SMA onde o campo, os robôs e a bola são o ambiente, cada robô é controlado por um agente autônomo, que possui uma percepção parcial do ambiente e com base nesta percepção e em seu papel no sistema pode executar um conjunto de ações sobre o ambiente alterando o estado do mesmo. A percepção de um agente resulta da visão parcial que o mesmo tem do ambiente, a área de visão é composta por um semicírculo frontal ao agente com centro no mesmo e ângulo de 90°. A visualização de um objeto, assim como a identificação de suas características, depende não apenas do mesmo encontrar-se dentro do semicírculo, mas também da distância deste ao agente que busca visualizá-lo. Quanto ao conjunto de ações que um agente pode executar sobre o ambiente, este é compreendido por:

- chutar a bola em uma determinada direção com uma determinada potência;
- deslocar-se para um determinado ponto com uma determinada velocidade;
- virar o corpo com um determinado momento;
- virar o pescoço com um determinado momento.

É necessário salientar que os agentes que compõem o sistema podem comunicar-se entre si. A figura 1 apresenta graficamente o que foi mencionado.

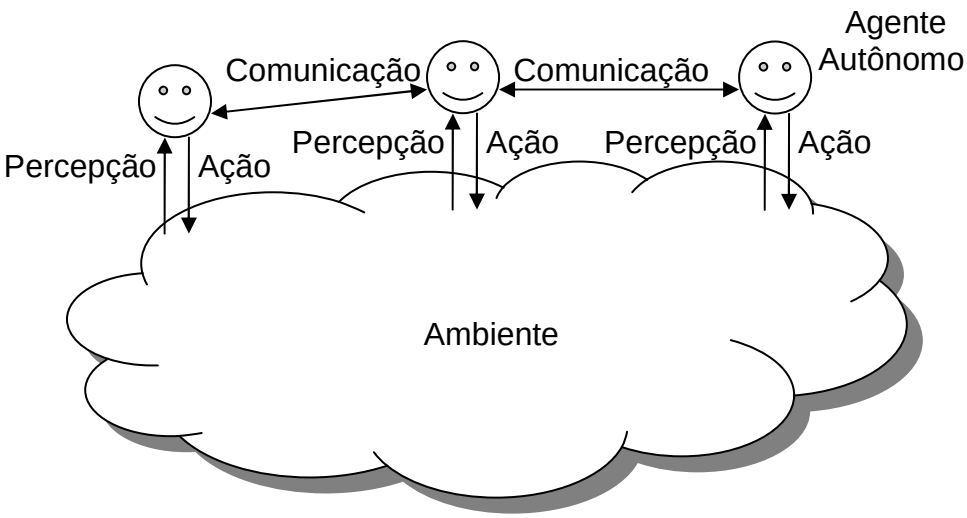


Figura 1 – Visão geral de um Sistema Multiagente

Uma partida de futebol é efetuada por duas equipes compostas por onze agentes cada, onde cada equipe possui como objetivo principal vencer a partida. A efetivação deste objetivo é buscada através do cumprimento de um conjunto de objetivos mais pontuais, denominados metas, que podem ser metas locais ou globais. Uma meta é definida como um estado desejado para o ambiente. Sendo uma meta local um estado do ambiente passível de ser atingido através de transformações no estado atual do ambiente em virtude da execução de uma ou mais ações por parte de apenas um agente sobre o ambiente. Já uma meta global é constituída por um estado do ambiente passível de ser atingido através de transformações no estado atual do ambiente em virtude da execução de ações por parte de um grupo de agentes sobre o ambiente.

Uma meta local ou global é realizada através da execução de um ou mais planos, que visam alterar o estado atual do ambiente para que este atinja o estado desejado, no caso em questão, a meta. Assim como as metas, os planos podem ser locais ou globais. Planos locais envolvem ações de um único agente e a sincronização dessas ações com os estados atingidos pelo ambiente. Já os planos globais, envolvem ações de diferentes agentes e a respectiva sincronização dessas com os estados assumidos pelo ambiente.

Cada agente possui um conjunto de planos e um papel definido na execução de cada um destes planos. Determinado plano é executado por um agente ou uma equipe de agentes, quando através de uma comunicação entre estes, levando-se em consideração o estado do ambiente percebido pelos agentes envolvidos na comunicação, é feita a seleção de um objetivo realizável através da execução de determinado plano.

O presente trabalho se propõe a solucionar o seguinte problema: com base em uma análise dos dados obtidos através de visões globais do ambiente, classificar o estado do mesmo, identificando ações dos agentes sobre este e com base nestas informações gerar, simultaneamente com o andamento das transformações do ambiente, um modelo comportamental de um grupo de agentes imerso no ambiente, estabelecendo um formalismo para representação deste modelo. O ambiente em questão será o futebol entre robôs, o SMA será composto pelos times envolvidos no confronto e o modelo a ser obtido refletirá o comportamento do time oponente.

Neste ponto torna-se importante a definição de como é constituído um modelo comportamental do time oponente. Conforme apresentado em (LINDER; COSTA, 2007), tal modelo se constitui da caracterização das jogadas do adversário, sendo que uma jogada é descrita pelo conjunto de interações (passes de bola) e comportamentos (conduções de bola) que ocorrem desde a cobrança de lateral, tiro de meta, roubada de bola, etc. até a saída da bola, a perda da posse de bola pelo oponente ou a efetuação de um gol, situações estas refletidas por estados do ambiente.

Mais especificamente, o método proposto visa identificar os planos executados pelos agentes que constituem o time oponente no decorrer de uma partida de futebol entre equipes de robôs através da análise do posicionamento dos robôs e da bola no campo em cada cena que constitui a partida.

As visões globais do ambiente podem ser obtidas por uma câmera posicionada sobre o campo em uma partida real (figura 2), no entanto, visando eliminar a necessidade da aquisição e pré-processamento das imagens do campo, para obtenção das coordenadas dos robôs e da bola, foi feita a opção pela utilização da

Liga Simulada 2d da RoboCup (LSR) como domínio de aplicação para o método proposto. Sendo que, a LSR reflete com fidelidade os desafios, restrições e características apresentadas em partidas reais. A figura 3 apresenta um quadro de uma partida da LSR.

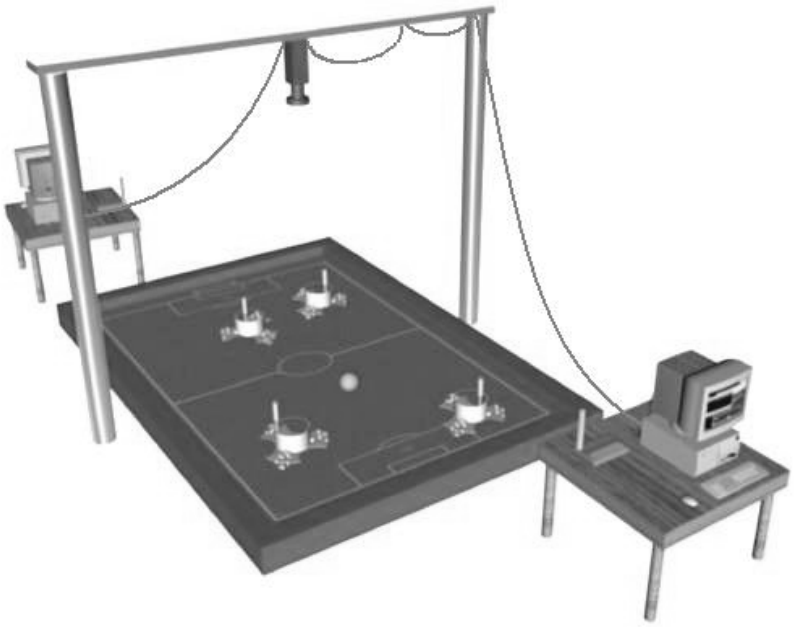


Figura 2 – Visão global de uma partida de futebol entre robôs F-180

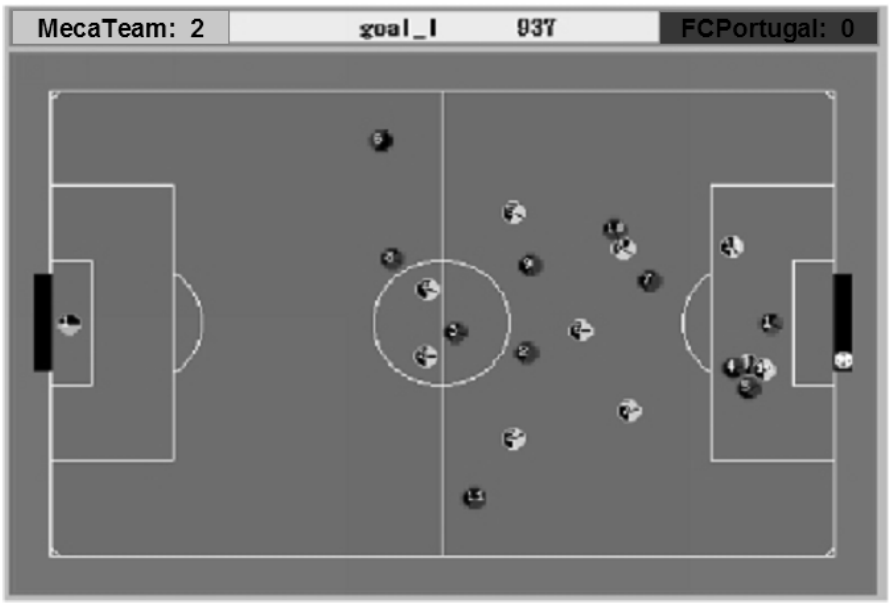


Figura 3 – Visão global de uma partida da LSR

Em partidas de futebol da LSR, as visões globais do campo são fornecidas, no decorrer do jogo, pelo *soccerserver* (servidor que controla a simulação) a um agente especial denominado treinador, o qual se utilizará do método proposto para, com base nas visões globais do ambiente, gerar o modelo comportamental do sistema em questão, processo este representado na figura 4.

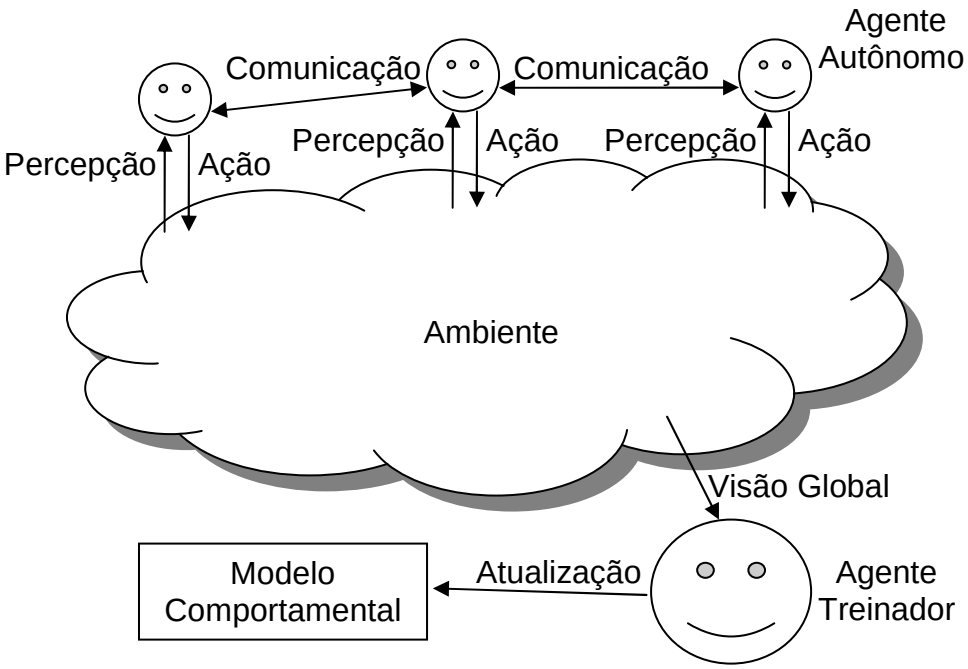


Figura 4 – Processo de aquisição da visão global e geração do modelo comportamental

(see_global 2962 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -31.2901 -17.0691 -0.742522 0.027956) ((p "Oxente_2005_V1_2" 1 goalie) -40.9135 -9.07116 -0.0154391 -0.355873 -94 53) ((p "Oxente_2005_V1_2" 2) -30.5075 5.82168 -0.39628 -0.0239665 178 89) ((p "Oxente_2005_V1_2" 3) -32.0742 -8.81491 -0.0288051 -0.0534979 159 90) ((p "Oxente_2005_V1_2" 4) -29.3556 -3.30352 -0.361573 -0.0730345 -175 77) ((p "Oxente_2005_V1_2" 5) -24.0313 -15.2067 -0.383974 -0.0432638 -176 11) ((p "Oxente_2005_V1_2" 6) -14.3533 -15.2285 -0.407853 -0.027028 -175 2) ((p "Oxente_2005_V1_2" 7) -13.5114 6.95488 -0.446754 -0.00138938 -175 48) ((p "Oxente_2005_V1_2" 8) -17.1422 -4.27836 -0.383967 -0.0258243 178 44) ((p "Oxente_2005_V1_2" 9) 2.49296 -4.51433 -0.0961853 0.00909529 -178 19) ((p "Oxente_2005_V1_2" 10) 7.82455 14.2512 -0.387435 -0.0340119 -173 31) ((p "Oxente_2005_V1_2" 11) 7.56249 -22.1956 1.70083e-06 -4.65341e-07 -158 -27) ((p "UvA_Trilearn" 1 goalie) 49.4226 -1.12598 -5.93373e-12 7.5306e-10 -89 -79) ((p "UvA_Trilearn" 2) 3.44512 -21.6703 0.000646739 -0.0052637 56 31) ((p "UvA_Trilearn" 3) 4.68253 -3.89137 -0.0245115 0.002321 -179 20) ((p "UvA_Trilearn" 4) -5.7414 -3.89419 -0.656383 0.0281255 180 27) ((p "UvA_Trilearn" 5) 2.1002 5.28163 -0.468919 -0.163207 -167 20) ((p "UvA_Trilearn" 6) -16.9918 4.27092 -0.688767 -0.0633894 -173 49) ((p "UvA_Trilearn" 7) -16.2194 -16.1343 -0.224155 -0.0346582 -177 -34) ((p "UvA_Trilearn" 8) -11.4534 4.44237 -0.325718 0.142511 173 55) ((p "UvA_Trilearn" 9) -23.7847 -9.00859 -0.365057 -0.016432 -172 4) ((p "UvA_Trilearn" 10) -30.0083 -17.2736 -0.625168 0.033226 174 -70) ((p "UvA_Trilearn" 11) -26.0363 15.7868 -0.392629 0.025744 -176 77))

Figura 5 – Exemplo de uma visão global fornecida pelo *soccerserver*

A figura 5 contém uma mensagem enviada pelo *soccerserver* para o agente treinador, contendo a descrição do ambiente em um determinado ciclo de simulação, através de um formalismo Quadros (MINSKY, 1975) para representação de conhecimento.

Alguns comentários podem ser feitos com relação às informações contidas na Mensagem de Descrição do Ambiente (MDA) apresentada na figura 5 com base no formalismo utilizado para representação de conhecimento:

- A mesma refere-se a um quadro que descreve um instante da partida;
- Como a *string* representa uma mensagem enviada pelo *soccerserver* ao agente treinador esta começa com a identificação de que tipo de mensagem se trata, no caso, *see_global*, ou seja, visão global;
- Logo em seguida, é fornecida a indicação de a qual ciclo de simulação a descrição do quadro representa, no caso, representa a descrição do ambiente no ciclo 2962;
- Seguindo esta informação são fornecidas, respectivamente, as coordenadas do centro do gol à esquerda, à direita, e as coordenadas da bola e dos jogadores;
- Para detalharmos como são descritas as coordenada dos objetos envolvidos na cena discutiremos o trecho ...(b) -31.2901 -17.0691 -0.742522 0.027956..., o qual contém informações referentes ao posicionamento da bola no quadro descrito, (b) indica que as coordenadas que seguem descrevem a localização do objeto bola, sendo -31.2901 a coordenada x, -17.0691 a coordenada y, -0.742522 o valor de Δx e 0.027956 o valor de Δy, as posições dos objetos no campo não são dadas em valores absolutos de suas coordenadas x e y, e sim, são fornecidas coordenadas de uma área retangular onde o objeto encontra-se, logo, as coordenada x e y referem-se a posição de uma das extremidades do retângulo, sendo Δx e Δy coordenadas relativas à x e y necessárias para a obtenção das demais extremidades do retângulo;

- Para finalizarmos a explanação sobre o tipo de entrada para o método, analisaremos agora o seguinte trecho: ...(p "Oxente_2005_V1_2" 6) -14.3533 -15.2285 -0.407853 -0.027028 -175 2)..., o qual contém informações referentes ao jogador número 6 do time "Oxente_2005_V1_2", sendo (tipo_do_objeto nome_do_time número_do_jogador) x y Δx Δy ângulo_do_corpo ângulo_do_pescoço).

Elevando o nível de abstração para complementar a compreensão do formalismo utilizado para representação do conhecimento na MDA, obtemos o diagrama apresentado na figura 6.

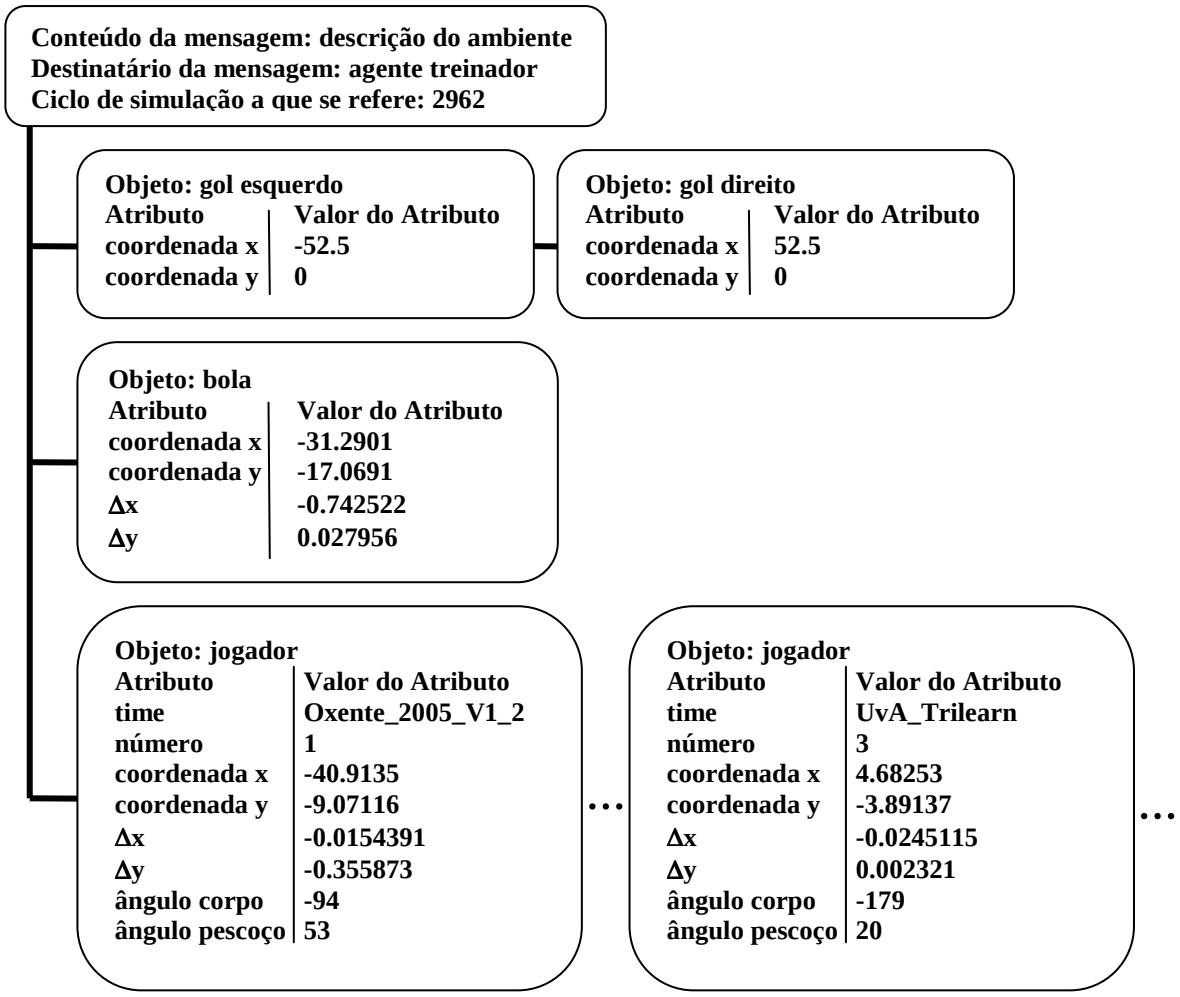


Figura 6 – Diagrama que resume o formalismo para representação do conhecimento presente na MDA

Em resumo a solução do problema exposto é obtida através da execução dos seguintes passos:

1. Para cada ciclo de simulação cuja respectiva MDA é enviada ao agente treinador é identificado o estado do ambiente;

2. Com base na classificação do estado do ambiente são identificadas as ações e/ou comportamentos executados pelos agentes presentes no sistema;

3. As ações e/ou comportamentos identificados são estruturados passando a constituir o modelo comportamental do time oponente.

Neste ponto, torna-se importante salientar a aplicabilidade da exploração do modelo comportamental do oponente em um ambiente competitivo. Com base na análise do histórico comportamental do adversário pode-se conjecturar a respeito de seus movimentos futuros, com o objetivo de torná-los ineficientes ou inviabilizá-los. Outra possibilidade de exploração do modelo comportamental do oponente seria a identificação de características positivas no comportamento do mesmo e absorção destas características.

3. TRABALHOS CORRELATOS

Em virtude dos desafios envolvidos em se analisar o comportamento inteligente que emerge a partir das interações entre os agentes, grupos de pesquisadores se dedicaram a este tema, produzindo contribuições como as evidenciadas em (NAIR; TAMBE, 2002), (SUKTHANKAR; SYCARA, 2005) e (RILEY; VELOSO, 2001). Trabalhos foram desenvolvidos visando a obtenção automática do modelo comportamental de sistemas multiagentes através da observação dos mesmos, como, por exemplo, (SUKTHANKAR; SYCARA, 2005). Contudo, para a obtenção dos trabalhos correlatos não foram analisados apenas trabalhos que se propunham a obter um modelo comportamental do SMA, uma vez que em determinados trabalhos, como em (NAIR; TAMBE, 2002), que se propõem a analisar o comportamento de SMA´s, os dados que são entrada para as análises podem constituir modelos comportamentais parciais do SMA, obtidos através de um pré-processamento que antecede a análise do SMA.

Uma dificuldade que se apresentou durante a busca de trabalhos correlatos foi a utilização de títulos e palavras-chaves, que podem ser associados à obtenção de um modelo comportamental de SMA´s, em trabalhos que se propunham a construção de ambientes virtuais de simulação. Um exemplo do que foi mencionado é encontrado no trabalho apresentado em (SILVA, 2003).

A seguir serão discutidos trabalhos correlatos ao tema desta dissertação, detalhando-se as vantagens e desvantagens de cada um. É interessante observar que alguns dos trabalhos que serão tratados se utilizam da LSR como domínio de aplicação.

3.1. Técnicas Automatizadas para Analisar Equipes de Agentes

Baseando-se na necessidade de se desenvolver técnicas automatizadas para analisar equipes de agentes, um grupo da University of Southern Califórnia desenvolveu um assistente automatizado que realiza análises de equipes de agentes, a LSR foi utilizada como domínio de aplicação para a efetuação das análises. O assistente desenvolvido é denominado ISAAC (RAINES; TAMBE;

MARSELLA, 2000), suas análises são feitas sobre jogos realizados. ISAAC requer pouco conhecimento do domínio, ele extrai dados da própria partida que está analisando, através do arquivo de *log* que contém o histórico das ações executadas pelos agentes em cada ciclo de simulação, e utiliza ferramentas de aprendizagem indutiva para fazer suas análises, mais especificamente, de árvores de decisão.

O ISAAC faz uma aproximação para o problema da análise de times utilizando uma análise *bottom-up* (do específico para o genérico) dos rastros de comportamento do time. Inicialmente são derivados padrões de comportamento bem sucedidos e mal sucedidos, como, por exemplo, são identificados padrões de chutes a gol bem e mal sucedidos. Tendo se definido, por exemplo, chutes a gol como uma parte esclarecedora para analisar a história global, deve-se determinar quais características podem ser úteis para se classificar o sucesso ou fracasso em um chute a gol. Exemplos de características são apresentados no quadro 1.

Velocidade da bola;
Distância do chutador ao gol;
Ângulo do chutador;
Distancia do oponente mais próximo do chutador;
Ângulo do adversário mais próximo do centro do gol;
Posição extrapolada da linha do gol (posição onde a bola passaria a linha do gol baseado na trajetória inicial, medida como distância do centro do gol;
Número de adversários entre o chutador e o gol.

Quadro 1 – Exemplo de características

Algumas destas características não são facilmente adquiridas e, além disso, quanto mais características forem analisadas maior será a complexidade computacional do processamento. Experiências anteriores confirmaram que, a análise de características irrelevantes diminui a precisão das regras produzidas pelo mecanismo de aprendizagem indutivo (CARUANA; FREITAG, 1994).

Uma vez determinadas quais características serão utilizadas na análise e quais os pontos da história (ou casos) serão examinados, uma árvore de decisão indutiva

(RUSSELL; NORVIG, 1995) será usada para essa análise, gerando regras como as apresentadas no quadro 2.

Distância do oponente mais próximo > 218 Ângulo do oponete mais próximo em relação ao centro do gol <= 8.981711 Ângulo com relação ao centro do campo > 40.77474 → classe sem gol

Quadro 2 – Exemplo de regras geradas

Através do conjunto de regras geradas é feita a análise do time, identificando quais características foram responsáveis pelo insucesso ou sucesso na realização de determinado evento. Considerando nosso objetivo proposto, ou seja, com base em uma análise dos dados obtidos através de visões globais do ambiente, classificar o estado do mesmo, identificando ações dos agentes sobre este e com base nestas informações gerar, simultaneamente com o andamento das transformações do ambiente, um modelo comportamental de um grupo de agentes imerso no ambiente, ISAAC não se apresenta como uma possível solução para nosso problema. Inicialmente, ISAAC não efetua suas análises simultaneamente com o decorrer da partida e não gera um modelo comportamental e sim classifica conjuntos de eventos, executados por um agente, em bem e mal sucedidos extraindo fatores que possam ter influenciado nestes desfechos. Além disto, a deficiência principal apresentada por ISAAC, tendo como base nosso objetivo, é que este se utiliza dos arquivos de *log* gerados pelo servidor ao término de cada partida, os quais contêm a seqüência de todas as ações executadas pelos agentes em cada ciclo de atualização do jogo. Isto gera uma grande lacuna, uma vez que objetivamos observar o andamento de um jogo, obtendo visões globais do mesmo e identificando com base nestas, as ações executadas pelos agentes.

Com a continuação das pesquisas, o ISAAC foi aperfeiçoado (NAIR; TAMBE, 2002), passando a efetuar análises em três níveis distintos, destinados a analisar:

1. Comportamentos individuais de agentes;

2. Interações de um pequeno grupo do sistema multiagente, considerado como fundamental para a obtenção de sucesso ou fracasso da equipe;

3. Tendências de comportamentos globais de equipes.

O primeiro nível de análise já foi discutido. A análise, desenvolvida no segundo nível, é feita através de uma identificação e classificação de padrões, cada um com uma probabilidade de ocorrência, os padrões são caracterizados por um autômato finito obtido com base nas interações entre os agentes. Este processo pode ser mais facilmente entendido quando aplicado no domínio da RoboCup.

Devemos nos recordar que um gol pode ser a meta de um plano, ou seja, o objetivo de uma jogada de ataque realizada pelos agentes, a qual é composta por um conjunto de ações ou comportamentos a serem realizados pelos agentes. Não sendo apenas o último chute que é responsável pelo gol, mas, sim o conjunto de ações de diferentes agentes que interagiram antes do gol. Logo, gera-se um autômato finito que representa este conjunto de interações, o autômato gerado por ISAAC é definido como: $G = \langle S, A, L, a_0, g \rangle$ onde,

S, é um conjunto de símbolos s_1, s_2, ..., s_M encontrados;

A, é um conjunto de estados (ou nodos) a_1, a_2, ..., a_N onde cada estado a_i consiste do seguinte:

a_i^s, o símbolo reconhecido pelo estado a_i. $a_i^s \in S$;

i, o número do estado, o qual unicamente identifica o estado;

a_0, o estado inicial, onde o número do estado é igual a 0 (zero) e $a_0^s = \varnothing$;

g, o símbolo da meta onde $g \in S$. O estado que reconhece g é referenciado como estado meta, ou seja, estado objetivo;

L, um conjunto de ligações dirigidas l_1, l_2, ..., l_p onde cada ligação l_i consiste do seguinte:

l_i^f , número do estado fonte da ligação;

l_i^t , número do estado destino da ligação;

l_i^Σ , número de vezes que a ligação foi atravessada enquanto adicionadas novas seqüências de símbolos (freqüência de ligações).

As freqüências dos padrões aprendidos pelo autômato finito são as freqüências das ligações que tem como estado fonte o estado inicial, i. e., valores de l_i^Σ para cada ligação l_i onde $l_i^f = a_0^s = 0$. Sendo possível obter a distribuição de probabilidade dos padrões de interações através de suas freqüências de ocorrência. As distribuições de probabilidades obtidas constituem a análise feita por ISAAC.

Por fim a análise das tendências de comportamentos globais de equipes, assim como na análise dos comportamentos individuais de agentes, utiliza uma aprendizagem por árvore de decisão, porém ao invés de se analisar pontos (casos) da história como sucesso intermediário ou fracasso, são analisadas características globais que conduzem ao resultado final sobre um plano completo. Estes resultados podem ser classificados como sucesso, fracasso, empate, etc. Neste caso também se faz necessário um especialista no domínio da aplicação para escolher estas características e prover as classes dos resultados finais.

Novamente para um melhor entendimento vamos nos valer do domínio da RoboCup para elucidar esta análise. Para a criação das regras de por que times (equipes) tiveram sucesso ou falharam no comprimento de planos pré-estabelecidos ISAAC necessitou inicialmente de dados fornecidos por um especialista no domínio, que fornecerá informações como as características contidas no quadro 3 e também fornecerá as classes de regras como as apresentadas no quadro 4. Com base nestas informações ISAAC cria regras que serão utilizadas na análise, o exemplo de uma regra é apresentado no quadro 5. Posteriormente estas regras serão usadas para geração de um resumo em linguagem natural que descreverá o comportamento global dos times.

1. Tempo de posse: Porcentagem de tempo que a bola estava em posse do primeiro time;
2. Bola em campo oponente: Porcentagem de tempo em que a estava no lado do campo do time oponente;
3. Defesa foi evitada: Número de vezes que o último defensor do primeiro time foi evitado;
4. Defesa oponente evitada: Número de vezes que o último defensor do time oponente foi evitado;
5. Distância mantida entre os próprios jogadores: distância comum mantida entre jogadores do primeiro time;
6. Distância mantida entre jogadores do time oponente: distância comum mantida entre jogadores do time de oponente;
7. Distância de último defensor: distância comum do último defensor do primeiro time ao gol;
8. Distância do último defensor oponente: distância comum do último defensor do time oponente ao gol.

Quadro 3 – Exemplo de características globais

1. Goleada (uma vitória através de 5 gols ou mais de diferença);
2. Vitória (uma vitória de 2 a 4 gols de diferença);
3. Vitória apertada (uma vitória com apenas um gol de diferença)
4. Empate;
5. Derrota (uma derrota através de apenas um gol);
6. Derrota moderada (uma derrota através de 2 a 4 gols de diferença);
7. Derrota vergonhosa (uma derrota através de 5 ou mais gols de diferença).

Quadro 4 – Exemplo de classes de regras globais

Tempo de posse > 52%
Bola em campo oponente >69%
Defesa foi evitada < 5
Defesa oponente evitada > 30
Distância mantida entre jogadoras do time oponente > 15m
-> Classe goleada

Quadro 5 – Exemplo de regra de time para goleada

Apesar dos avanços apresentados por ISAAC em suas análises, quando contraposto com nosso objetivo percebe-se que ainda mantém algumas deficiências. No terceiro nível de análise efetuado por ISAAC, o qual aproxima-se mais do nosso objetivo, não é efetuada uma análise comportamental de uma equipe em específico, e sim são identificadas tendências comportamentais, ou seja, é feita uma análise do por que times têm sucesso ou falham em um determinado contexto. Porém o principal problema continua sendo a questão deste se utilizar do arquivo de *log*, gerado pelo servidor ao término de cada partida, contendo as ações executadas pelo agentes, como entrada para suas análises. Fato este que evidência a necessidade de se propor um método que propicie a caracterização das ações através da observação do SMA, já que no problema que nos propomos a resolver não partiremos das ações e sim teremos que inicialmente caracterizá-las através da observação das mudanças ocorridas no estado do ambiente.

No entanto, é importante salientar que o autômato finito constante no segundo nível de análises efetuadas por ISAAC, serviu como base para a elaboração do método proposto em nosso trabalho, já que este se propõe a representar uma seqüência de interações entre agentes.

3.2. Metodologia para Reconhecer, Representar e Registrar Comportamentos Executados por uma Equipe de Agentes

Em (SUKTHANKAR; SYCARA, 2005) é proposta uma metodologia para reconhecer, representar e registrar comportamentos executados por uma equipe de jogadores humanos em um Torneio Fictício de Operações Militares em Terreno Urbano (OMTU). Para monitorar diretamente o desempenho de jogadores humanos, desenvolveu-se uma versão de Torneio Fictício que registra a posição e orientação no tempo de todos os membros da equipe, como eles participam em um contexto simulado de uma equipe de ataque que se move por uma área urbana em OMTU. O reconhecimento de comportamento é executado *off-line* usando um jogo de modelos de Markov (FINE ET AL, 1998) escolhidos em sucessões de movimentos curtos que são traduzidos em uma armação de referência canônica; o modelo de comportamento com a probabilidade mais alta para uma determinada sucessão é

identificado como correto. O trabalho pressupõe que um reconhecimento *off-line* preciso de comportamentos de time é uma condição prévia importante para construir ambientes de treinamento virtuais para tarefas de trabalho de equipe.

Para obtenção de resultados foi utilizado o seguinte procedimento:

1. Pares de jogadores humanos usando uma *interface* de jogo de Torneio Fictício modificada manipularam “bots” (uma espécie de agente não autônomo) por um pequeno ambiente urbano;

2. Depois de uma breve familiarização inicial com o mapa e trabalhando junto como um time, os jogadores foram instruídos para executar uma sucessão particular de comportamentos de time. Apesar de ter sido utilizado como ambiente de aplicação o Torneio Fictício é interessante salientar que as instruções executadas não se restringiam às praticadas no ambiente em questão e sim refletiam sucessões de OMTU comumente utilizadas na realização de manobras reais;

3. Usando a versão modificada de Torneio Fictício, rastros dos comportamentos dos jogadores foram registrados em um arquivo de texto para análise *off-line*, ou seja, após o termino do torneio;

4. Os rastros de comportamento eram pré-processados para minimização, através da sobreposição de segmentos gerando uma amarração de referência canônica baseado no movimento do centróide do time;

5. *Offline* os rastros foram classificados automaticamente utilizando-se modelos de Markov;

6. Foram comparados os resultados do reconhecimento automático com uma versão de aquisição manual do rastro.

A idéia central do trabalho acima mencionado em muito se assemelha com nossa proposta, contudo o domínio de aplicação é muito particular e foram feitas muitas simplificações e adaptações ao domínio para que a modelagem pudesse ser efetuada com o método proposto.

3.3. Reconhecimento e Adaptação ao Comportamento da Equipe de Agentes Oponente

Baseado na premissa de que em domínios multiagentes com adversários e agentes cooperativos, os agentes do time deveriam ser adaptáveis ao ambiente e oponente atual, (RILEY; VELOSO, 2001) introduz um método *on-line* que proporciona planos para o time de agentes, os quais são gerados por um agente específico denominado "treinador", obtidos com base em oponentes específicos. O treinador é equipado com vários modelos de oponentes pré-definidos, sendo este capaz de selecionar rapidamente, no decorrer da partida, um dentre os diferentes modelos usando um algoritmo Bayes (HUANG ET AL, 2006), gerando um plano, para os agentes do time, adaptado ao adversário atual. O treinador usa uma Rede Temporal Simples (PELED; TSAY, 2005) para representar planos de time, como movimentos múltiplos coordenados entre os agentes, e procura um plano dependente-do-oponente para seus companheiros de time. Este plano é comunicado então aos agentes que o executam de forma distribuída. O sistema é implementado completamente em um domínio de futebol de robôs simulado, mais especificamente na LSR.

O trabalho desenvolvido por (RILEY; VELOSO, 2001) visa classificar e responder aos comportamentos dos oponentes ao longo do jogo, focalizando-se em responder efetivamente após situações de bola parada, o que é conhecido como situações de jogada ensaiada. Foram introduzidos vários planos de jogada ensaiada prefixados que realmente provêem grandes oportunidades para posicionar estrategicamente os componentes do time buscando minimizar o desempenho do time adversário. No artigo (STONE ET AL, 2000) é proposta a construção de jogadas ensaiadas adaptativas, ou seja, que mudam ao longo do jogo visando uma otimização do desempenho em função do comportamento do time oposto. O treinador compila a visão global necessária para identificar como o time oponente se comporta, comunicando um plano que será executado pela sua equipe. O agente treinador é equipado com vários modelos de oponentes pré-definidos, sendo estes representações probabilísticas de movimentos dos agentes oponentes. O processo ocorre da seguinte forma:

- Na ocorrência de uma situação de bola parada, por exemplo, devido a uma cobrança de lateral, o treinador, rapidamente, tira proveito do curto tempo disponível para criar um plano de jogada ensaiada para seu time. O plano é gerado através de uma procura em um espaço de planos utilizando uma função de avaliação que embute as prováveis predições do modelo do oponente atual. A geração do plano, além do reconhecimento do modelo do oponente, usa novamente o modelo para predizer o comportamento dos agentes oponentes;

- Além deste algoritmo de criação de plano, um desafio adicional desta pesquisa foi a seleção e uso de uma representação apropriada de plano. O treinador usa uma representação de Redes Temporal Simples que efetivamente captura as dependências temporais entre os passos de um plano, a sua maior contribuição é possibilitar saltos temporais nos registros esperados de execução das ações;

- Na situação de bola parada é gerado e entregue o plano pelo treinador aos membros do time, incluindo as informações necessárias para os agentes executarem e monitorarem de maneira completamente distribuída o plano do time;

- O treinador também observa a execução do plano gerado para atualizar a seleção de um modelo apropriado para o oponente atual.

O sistema desenvolvido em (RILEY; VELOSO, 2001) é capaz de classificar o comportamento do time oponente após situações de bola parada. No entanto, o treinador compila a visão global necessária para classificar como o time oponente se comporta após a ocorrência de uma bola parada, identificado apenas características necessárias para que se possa selecionar um dentre os vários modelos de oponentes pré-definidos, sendo estes representações probabilísticas de movimentos dos agentes oponentes.

Objetivando explicitar as características observadas em cada trabalho analisado, possibilitando uma comparação entre os mesmos e demonstrando assim o porquê

da necessidade da criação do método proposto, foi gerada a tabela a seguir, que sintetiza o processo de análise dos trabalhos correlatos.

Trabalho / Característica Observada	(RAINES; TAMBE; MARSELLA, 2000)	(NAIR; TAMBE, 2002) Nível 2	(NAIR; TAMBE, 2002) Nível 3	(SUKTHANKAR; SYCARA, 2005)	(RILEY; VELOSO, 2001)
Gera modelo comportamental do sistema	Não	Não	Não	Sim	Não
Gera modelo comportamental parcial do sistema	Não	Sim	Não	Sim	Não
Efetua análises com base na observação das transformações no estado do ambiente	Não	Não	Não	Sim	Sim
Efetua simplificações no ambiente para obtenção do modelo comportamental do sistema	Não	Não	Não	Sim	Não
As análises ocorrem simultaneamente com o decorrer das transformações do ambiente	Não	Não	Não	Não	Sim
O sistema modelado é simples quando comparado ao sistema que se propõe modelar nesta dissertação	Não	Não	Não	Sim	Não

Tabela 1 – Síntese da análise dos trabalhos correlatos

4. MOMCoTO – Método para Obtenção do Modelo Comportamental do Time Oponente

No capítulo 2 foi esclarecido que tipo de dado será entrada para o método e caracterizada a constituição de um modelo comportamental. Definiremos agora que formalismo será utilizado para representar o modelo gerado.

Como pode ser observado no capítulo anterior, foram utilizados alguns formalismos para a representação dos modelos comportamentais dos SMA analisados, como: Redes Temporais Simples e Autômatos Finitos. Levando-se em consideração além das características do sistema a ser modelado, discutidas anteriormente, como transições entre estados do sistema ocasionadas pela execução de ações ou interações entre agentes, foi ponderado para a definição do formalismo, que o método para modelagem comportamental proposto neste trabalho visa ser o embrião para novos projetos que virão a complementá-lo. Logo, em função de futuras sofisticações no modelo comportamental gerado, através da consideração de mais variáveis envolvidas no sistema como, por exemplo, marcos temporais, probabilidades de ocorrência e paralelismo entre ações identificadas, o formalismo a ser escolhido necessita ser robusto o suficiente para que não seja necessário substituí-lo com os avanços do método. Contudo, outro detalhe relevante é que além de um alto poder de representatividade o formalismo a ser selecionado deve ser capaz de representar o modelo comportamental gerado pelo método, em sua atual concepção, sem tornar a representação do modelo complexa. Os formalismos utilizados nos trabalhos correlatos não contemplam tais características.

Tendo em vista o que foi mencionado, optou-se pela utilização de uma Rede de Petri (RdP) para representar as jogadas identificadas do oponente, devido ao potencial de representatividade da mesma, que tem sido amplamente utilizada para modelagem de sistemas concorrentes, temporizados e/ou estocásticos, como demonstram: (GIRAULT; VALK, 2002), (DESEL, 2000), (MACIEL, 1996), (MURATA, 1989), (PETERSON, 1981) e (PETERSON, 1977). Posteriormente, será evidenciado que, apesar do grande potência de representatividade, uma RdP é capaz de exprimir o

modelo comportamental gerado pelo MONCoTO, em sua atual concepção, sem tornar a representação do modelo complexa.

4.1. Conceitos básicos sobre RdP

Neste ponto torna-se necessária, para discussões posteriores, a definição dos seguintes conceitos:

Segundo (PENHA ET AL, 2004), uma RdP é composta pelos seguintes elementos:

- Lugares ou *Places*: representam uma condição, uma atividade ou um recurso.
- Fichas, Marcas ou *Tokens*: representam o estado de um sistema.
- Transições: representam um evento.
- Arcos: indicam os lugares de entrada ou saída para as transições.

Estes elementos podem ser representados graficamente como na figura 7.

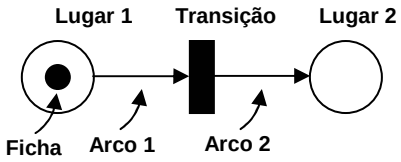


Figura 7 – Elementos de uma Rede de Petri

Portanto, podemos definir uma Rede de Petri como R = (P, T, AE, AS), onde:

- P = {P1, P2, ..., Pm} é um conjunto de lugares.
- T = {T1, T2, …, Tn} é um conjunto de transições.
- P ∩ T = ∅ os conjuntos P e T são disjuntos.
- AE: P x T é o conjunto de arcos de entrada nas transições.

• AS: T x P é o conjunto de arcos de saída das transições.

Para a figura 7, temos:

• P = {Lugar 1, Lugar 2}

• T = {Transição}

• AE = {Lugar 1, Transição}

• AS = {Transição, Lugar 2}

4.2. Constituição da RdP que representa o modelo comportamental

Desta forma, ao fazer a opção pela utilização de uma RdP torna-se necessário definir qual o conjunto de lugares e transições que constituirão a rede. Para tornar a identificação dos lugares suficiente para que se faça uma alusão direta desta com a condição a qual representa, ou melhor, com o estado em que o sistema deve encontrar-se, optou-se por identificar os lugares com uma estrutura composta pela concatenação:

- da localização em que a bola encontra-se;
- de que time tem posse da bola;
- e com que jogador deste time a bola encontra-se.

Todavia, se for considerada a posição absoluta dos objetos como uma descrição precisa de sua localização, existirá uma infinidade de localizações possíveis para os jogadores e a bola no campo, por exemplo, quando o jogador 9 do time oponente estiver na posição (4.7894, 67.3421) possuirá uma localização e quando este estiver na posição (4.7895, 67.3421) possuirá outra localização, o que geraria uma quantidade inadmissível de lugares na RdP. Em virtude, desta observação, ou seja, com o intuído de se tornar discreta a localização, o campo foi setorizado em quadrantes conforme apresentado na figura 8.

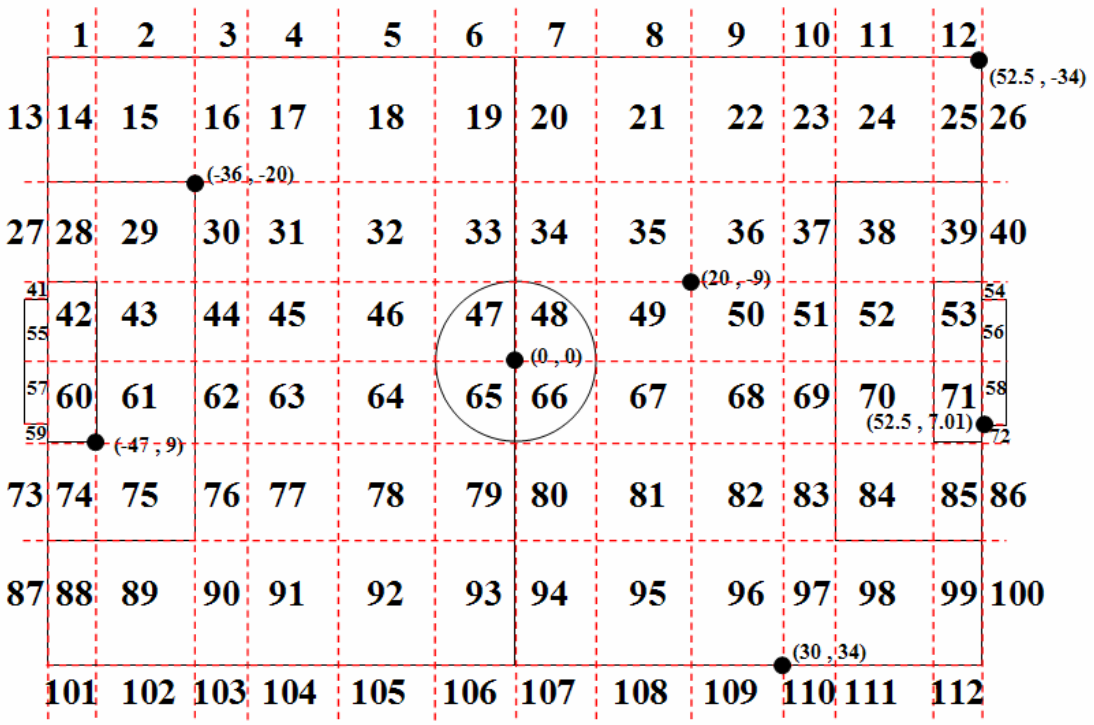


Figura 8 – Campo dividido em quadrantes

Esta divisão foi feita com base nas áreas já delimitadas nos campos de futebol (pequena área, grande área, etc.) e levando em consideração observações da dinâmica de jogos entre equipes de robôs autônomos. Pode-se verificar na figura 8 que devido à simetria do campo é necessária apenas a definição de sete pontos para a delimitação dos 112 quadrantes.

Baseado na divisão do campo em quadrantes, a identificação dos lugares da RdP passam a ser através de uma *string* resultante da concatenação de um inteiro, um caractere e outro inteiro. Onde:

- o primeiro inteiro representa o quadrante onde a bola encontra-se, logo este possui valores entre 1 e 112;
- o caractere pode possuir o valor ‘t’ ou o valor ‘o’, quando a bola estiver em posse do time oponente o valor será ‘o’ caso contrario o valor será ‘t’;
- o último inteiro refere-se ao número do jogador que tem posse da bola, logo este pode variar de 1 à 11.

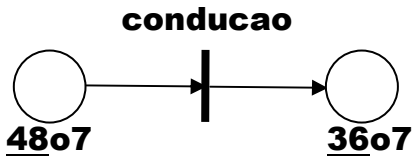
Por exemplo, no caso em que a bola esteja no quadrante 89 em posse do jogador oponente número 11, a identificação do lugar que represente tal condição na rede será 89o11. Uma ressalva deve ser feita, sempre que a bola se encontrar fora dos limites do campo, ou seja, em um dos seguintes quadrantes: 1 a 12, 13, 26, 27, 40, 41, 54, 55, 56, 57, 58, 59, 72, 73, 86, 87 ou 100 a 112, a identificação do lugar é feita apenas pelo número do quadrante em que a bola se encontra salvo quando esta se encontrar nos quadrantes 55 ou 58 quando o lugar passa a ser identificado por gol_ld (gol do lado direito) ou quando esta se encontrar nos quadrantes 56 ou 57 quando o lugar passa a ser identificado por gol_le (gol do lado esquerdo). Para exemplificar o que foi dito, no caso da bola encontrar em jogo em posse do time oponente com o jogador número 5 no quadrante 22 o estado do jogo será representado na RdP por “22o5”.

Quanto às transições que compõem a RdP, os eventos identificados são:

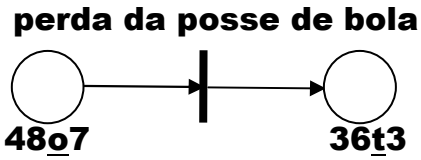
- **passe** – caracterizado pela troca de posse de bola entre jogadores do time oponente ou pela bola ter ultrapassado um dos limites laterais do campo, estando esta em posse do time oponente, a seguir exemplos;

- **chute** – caracterizado pela ocorrência de um gol do time oponente ou pela bola ter ultrapassado um dos limites do fundo do campo, estando esta em posse do time oponente, a seguir exemplos;

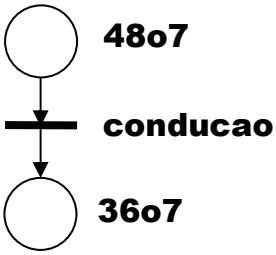
- **condução** – caracterizada pela variação no quadrante de localização da bola quando esta permanece em posse do mesmo jogador do time oponente entre o final de uma transição e o inicio de outra, a seguir exemplo;



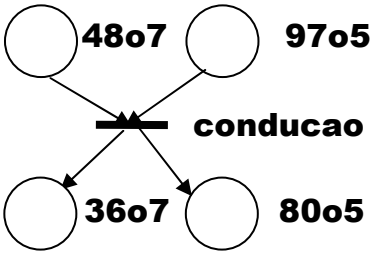
- **perda da posse de bola** – caracterizada pela troca de posse de bola entre o time oponente e o outro time, a seguir exemplo.



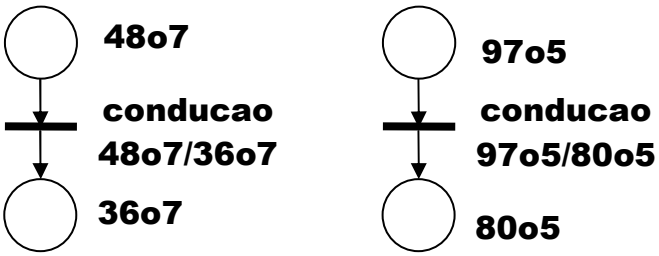
No entanto, alguns problemas podem surgir quando as transições definidas são identificadas apenas pelos seus tipos. Imagine a seguinte situação: O jogador oponente 7 está no quadrante 48 com a bola e efetua uma condução até o quadrante 36, o que seria representado por



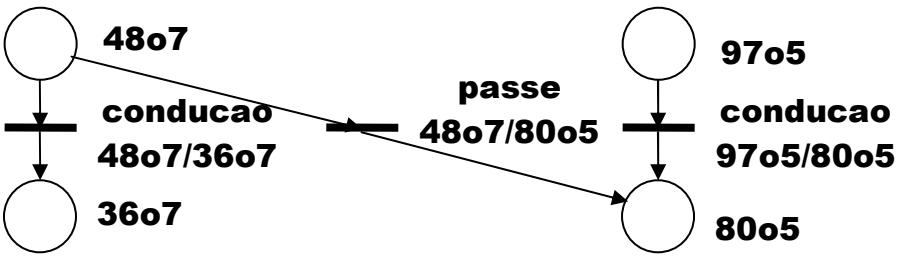
Agora vamos imaginar que em outro instante do jogo o jogador oponente 5 está no quadrante 97 com a bola e efetua um condução até o quadrante 80, o que seria representado na RdP já existente por



o que não representa de forma correta a ocorrência das duas situações anteriores. Pois, da forma em que a RdP encontra-se “48o7” e “97o5” são condições necessárias para a ocorrência do evento condução e nunca as duas serão satisfeitas simultaneamente, devido ao fato de existir apenas uma bola em jogo. Fato este, que também inviabiliza a possibilidade de após a ocorrência do evento condução o sistema encontrar-se simultaneamente nos estados “36o7” e “80o5”. Desta forma, uma decisão de projeto foi identificar as transições pelo seu tipo, lugar de origem e lugar de destino. Logo, as situações anteriormente mencionadas passam a ser representadas pela seguinte RdP:



Assim, se em outro instante do jogo, o jogador oponente 7 estiver novamente no quadrante 48 com a bola e efetua desta vez um passe para o jogador 5 que encontra-se no quadrante 80, a RdP passará a ter a seguinte representação gráfica:



4.3. Concepção do Método para Modelagem Comportamental

Após delinear o problema a ser tratado, especificar as entradas, definir e estabelecer o formalismo para descrição da saída gerada pelo método, será detalhada agora a forma como o método através das entradas gera as respectivas saídas.

O MOMCoTO é constituído por um módulo principal e um módulo secundário denominado Classificador, o qual é capaz de com base em uma visão global do SMA classificar em qual estado encontra-se o mesmo. O processo de modelagem ocorre da seguinte forma:

- O módulo principal recebe, uma a uma, as visões globais do campo e as envia para o módulo Classificador;
- O Classificador retorna ao módulo principal a indicação de em que estado o SMA encontra-se;
- Em posse desta informação o módulo principal verifica se foi dado início a uma jogada do oponente ou se existe uma jogada, do time oponente, em andamento; Caso uma jogada esteja em andamento, é feita a verificação, pelo módulo principal, se houve a finalização da jogada ou se ocorreu alguma das transições citadas anteriormente.
- No caso de alguma das situações anteriores ser constatada, o módulo principal, verifica se a RdP que representa o modelo comportamental do oponente deve ser atualizada, caso positivo a atualização é efetuada. A figura 9 apresenta a forma como é estruturado o MOMCoTO.

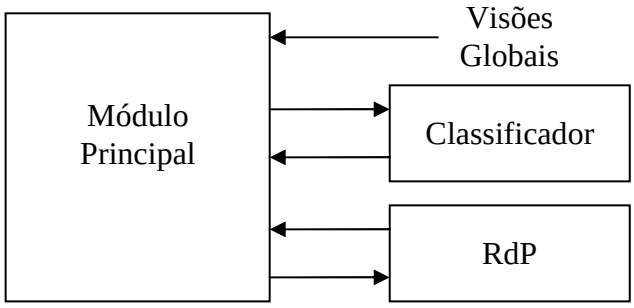


Figura 9 – Estrutura do MOMCoTO

Objetivando a identificação de estados necessários para a caracterização das transições, anteriormente mencionadas, o Classificador foi elaborado com a capacidade de, com base nas informações contidas em uma visão global do ambiente, identificar em qual dos seguintes estados o sistema encontra-se:

- Pré-inicial. Estado caracterizado em virtude da partida ainda não ter sido inicializada;
- Estado de posse de bola indefinido. Caracterizado em função do estado do jogo refletir uma possível troca de posse de bola entre jogadores;
- Saída de bola em posse do oponente. Estado caracterizado por ter ocorrido uma cobrança de lateral, saída de bola no meio de campo, etc. efetuada pelo time oponente;
- Desarme do oponente. Caracterizado pela perda da posse de bola por parte do time oponente;
- Troca de detentor de posse de bola. Caracterizado pela detecção da mudança de posse de bola entre jogadores do time oponente;
- Saída de bola em posse do time. Estado caracterizado por ter ocorrido uma cobrança de lateral, saída de bola no meio de campo, etc. efetuada pelo time;
- Fim do primeiro tempo. Caracterizado pelo termino do primeiro tempo da partida;
- Permanência de posse de bola com o time. Estado caracterizado em virtude do time continuar com a posse de bola;
- Permanência de posse de bola com jogador oponente. Caracterizado quando o mesmo jogador oponente continua com posse de bola;
- Gol do oponente. Estado caracterizado pela identificação da ocorrência de um gol efetuado pelo time oponente;

- Gol do time. Estado caracterizado pela identificação da ocorrência de um gol efetuado pelo time;
- Aquisição da posse de bola pelo time oponente. Caracterizado em virtude do time oponente ter adquirido a posse de bola;
- Ocorrência de lateral. Estado caracterizado pela bola ter ultrapassado os limites laterais do campo;
- Cobrança de tiro de meta. Caracterizado quando o estado do jogo reflete o instante de uma cobrança de tiro de meta;
- Estado transitório. Estado caracterizado quando após a ocorrência de um gol, do final do primeiro tempo ou da bola ter ultrapassado um dos limites laterais do campo, está ainda não retornou ao jogo.

Conforme mencionado, com base no retorno do Classificador o módulo principal caracteriza:

- O início de uma jogada do oponente;
- A finalização de jogada do oponente, que pode ocorrer em virtude de um desarme, da efetuação de um gol ou da bola ter ultrapassado um dos limites do campo;
- Um passe entre jogadores do time oponente;
- Uma condução de bola executada por um jogador do time oponente.

A classificação referida, feita através do retorno do Classificador, ou seja, com base no estado do SMA, reflete que caso deve ser tratado. Sempre que for classificado o estado “Pré-inicial” ou o “Estado transitório” nada deve ser feito, a não ser esperar por novas visões globais. No entanto para cada um das demais possibilidades de retorno uma determinada ação ou conjunto de ações deve ser executado, objetivando a atualização da RdP, para que a mesma venha a representar o modelo

comportamental do SMA. A seguir é descrita a forma como o modulo principal caracteriza os casos citados:

- O início de uma jogada do oponente é caracterizado pela identificação dos estados “Saída de bola em posse do oponente” ou “Aquisição da posse de bola pelo time oponente”;

- Já um passe ocorre quando identificado o estado “Troca de detentor de posse de bola”;

- O final de uma jogada é caracterizado quando uma jogada do oponente está em andamento e o SMA atinge em um dos estados: “Desarme do oponente”, “Fim do primeiro tempo”, “Gol do oponente”, “Ocorrência de lateral” ou “Cobrança de tiro de meta”;

- A identificação de uma condução ocorre sempre no instante que se caracteriza a ocorrência de um passe ou a finalização da jogada do oponente em andamento, o processo se dá da seguinte forma: sempre que um novo jogador do time oponente adquire posse de bola, no início de uma jogada ou no instante do recebimento de um passe, é armazenada a posição em que a bola encontrava-se neste instante. Posteriormente quando é identificado que este jogador não possui mais a posse da bola, em virtude deste ter passado a bola, ter efetuado um chute, da bola ter saído do campo, de ter ocorrido um desarme ou simplesmente ter acabado um dos tempos do jogo, é feita uma verificação se, no instante da ocorrência de um destes eventos mencionado, a bola encontrava-se em uma posição divergente da armazenada, caso positivo caracteriza-se a ocorrência de uma condução entre as respectivas posições.

No instante da identificação de cada uma das situações descritas anteriormente são inseridos os respectivos lugares e transições na RdP que representa o modelo comportamental do time oponente, caso estes estados ou transições ainda não se encontrem na RdP.

4.3.1. Exemplo de Construção de uma RdP Representando o Modelo Comportamental do Time Oponente

Para que se possa obter uma melhor compreensão do que foi mencionado, será feita a análise de uma situação hipotética onde se observará como é atualizada uma RdP representando o modelo comportamental de um SMA.

Em um determinado instante, de uma partida fictícia, a RdP apresentada na figura 10, representa o modelo comportamental do time oponente até o instante em que, estando a bola em posse do time, ou seja, o estado atual do sistema é classificado como “Permanência de posse de bola com o time”, é recebido pelo módulo principal uma visão global que ao ser analisada pelo Classificador indica que o sistema passou ao estado “Aquisição de posse de bola pelo time oponente”, refletido pela representação “78o4” , neste momento é armazenada a descrição do estado e a distância do jogador oponente 4 à bola para no futuro poder ser analisado se houve a ocorrência de condução. Neste instante é identificado o início de uma jogada do oponente, sendo então feita a verificação se o estado do jogo está representado na RdP constante na figura 10.

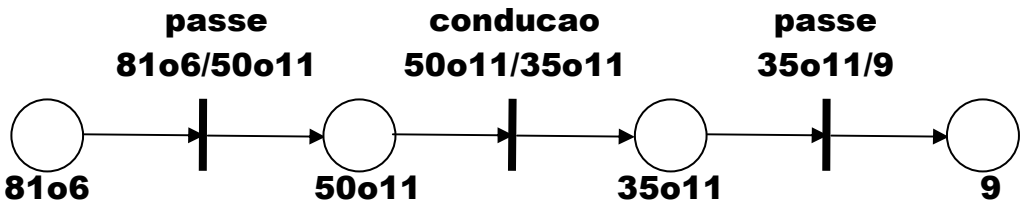


Figura 10 – Exemplo de RdP

É então constatado que a RdP deve ser atualizada, devido ao estado não estar representado na rede, este então é inserido passando a rede a ter a representação gráfica da figura 11.

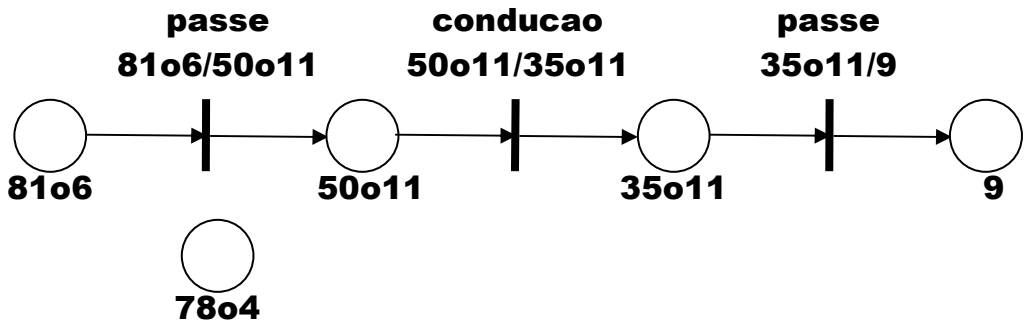


Figura 11 – Exemplo de RdP

Com a continuidade da partida são recebidas em seqüência duas visões que refletem, respectivamente, a classificação do sistema nos seguintes estados: “Estado de posse de bola indefinido” e “Troca de detentor de posse de bola”, neste momento, é analisado se houve condução por parte do jogador oponente número 4, sendo verificado se a posição da bola no estado em que o jogador oponente, que possuía a posse de bola, a adquiriu é igual à sua posição no estado armazenado para verificação da ocorrência de condução, no caso, é, esta condição representa a não ocorrência de uma condução. Após esta verificação, estando o sistema no estado representado por “81o6”, é armazenada a descrição do estado atual do sistema e a distância do jogador 6 à bola para no futuro poder ser analisado se houve a ocorrência de condução. Neste ponto, novamente é verificado se há necessidade de atualizar a RdP, é observando inicialmente se o estado “81o6” encontra-se na RdP, como este encontra-se nada é feito. Posteriormente é verificado se a transição “passe 78o4/81o6” esta presente na rede, como esta não se encontra é então inserida e a RdP passa a ter e representação constante na figura 12.

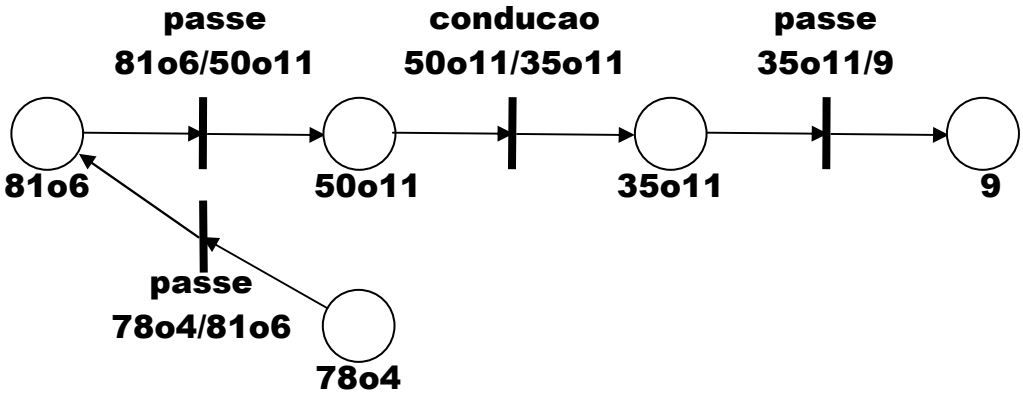


Figura 12 – Exemplo de RdP

Com a chegada de uma nova visão global ao módulo principal, é feita então a classificação do estado do sistema, que encontra-se no estado “Permanência de posse de bola com jogador oponente”. Neste instante é verificado se o jogador oponente, que possui a posse de bola, encontra-se mais próximo da bola do que no momento em que o mesmo adquiriu a posse de bola, no caso, é constatado que o jogador encontra-se mais próximo. Desta forma é armazenada a descrição do estado do sistema “68o6” e a nova distância do jogador à bola, para futura verificação da ocorrência de condução.

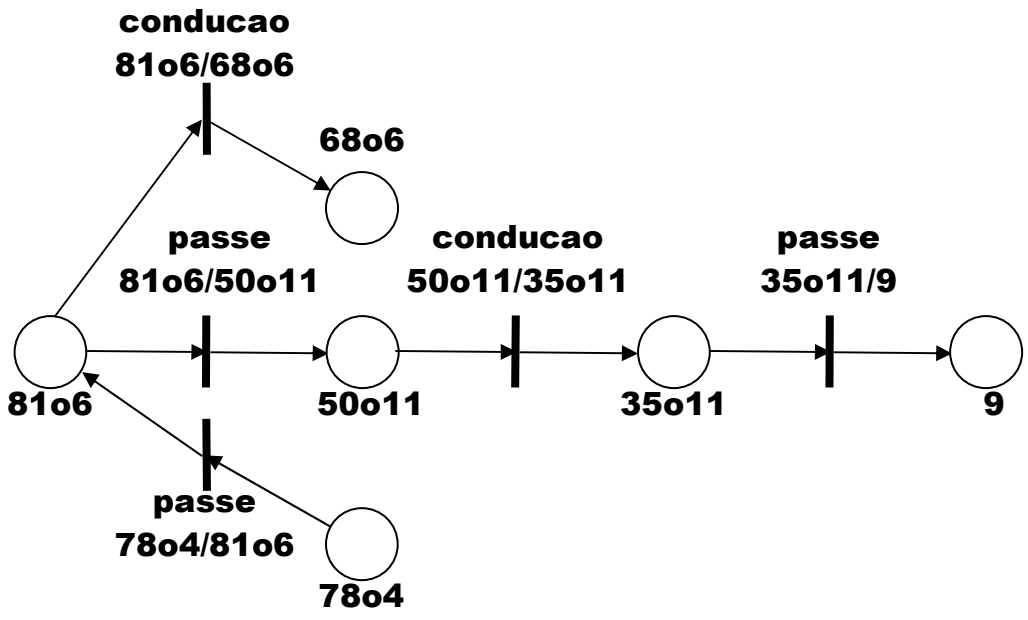


Figura 13 – Exemplo de RdP

Com a continuidade da partida uma nova seqüência de visões globais é analisada e constata-se, respectivamente, que o sistema passou pelo estado “Posse de bola indefinido” se encontrando no estado “Desarme do oponente”, caracterizando o final de uma jogada do time oponente. Neste momento, é analisado se houve condução, sendo verificado se a posição da bola no estado em que o jogador oponente, que possuía a posse de bola, a adquiriu é igual à sua posição no estado armazenado para verificação da ocorrência de condução, no caso, não é, esta condição representa a ocorrência de uma condução. Neste momento é verificado se existe a necessidade de atualização da RdP, analisando se o estado “68o6” pertence à rede, como não pertence é inserido e posteriormente verifica-se a presença na rede da transição “condução 81o6/68o6”, a qual também não esta presente, a mesma também é inserida. Desta forma a RdP passa a ter a representação apresentada na figura 13. Após estas atualizações da rede é verificado se o estado atual do jogo representado por “68t4” pertence à rede, como este não pertence é inserido e também é verificado se a transição “perda da posse de bola 68o6/68t4” pertence a rede, como esta também não está presente na rede a mesma é atualizada passando a ter a representação constante na figura 14.

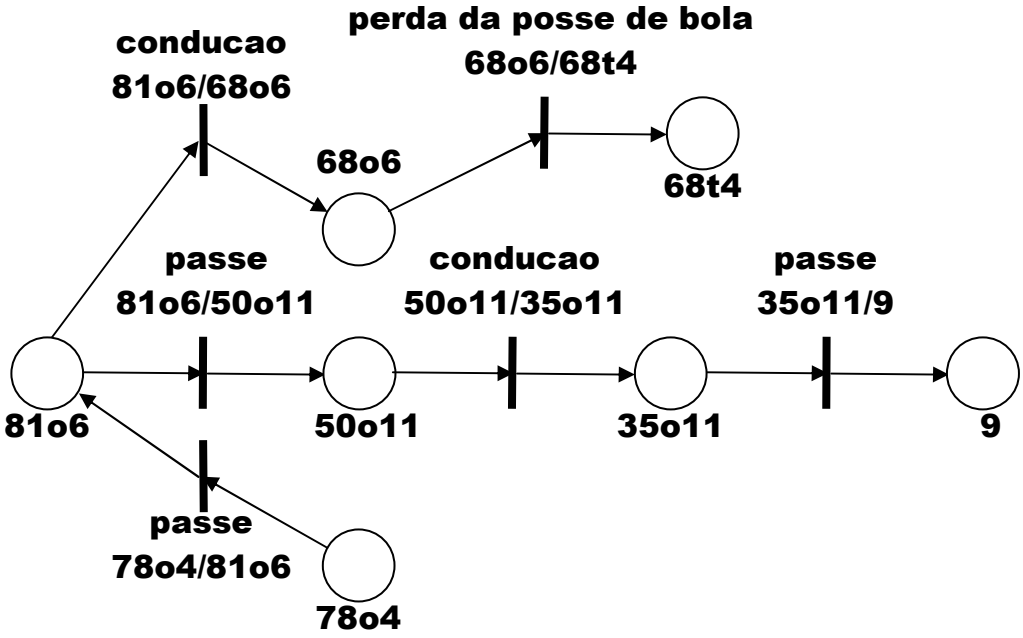


Figura 14 – Exemplo de RdP

Com o objetivo de esclarecer mais o processo descrito, pode ser visualizado na figura 15, a qual apresenta a matriz de entrada e a matriz de saída que representam a RdP apresentada na forma gráfica na figura 16, a mesma descreve o modelo comportamental do time oponente até a efetuação do primeiro gol de uma determinada partida analisada. As visões globais que descrevem este trecho do jogo, e foram entrada para o método proposto, são apresentadas no Apêndice I.

Matriz de Entrada

		48o9	48o8	34o7	33o10	46o9	32o9	16o10	29o10	44o9	44o6	gol_le
48o9/48o8	passe	1	0	0	0	0	0	0	0	0	0	0
48o8/34o7	passe	0	1	0	0	0	0	0	0	0	0	0
34o7/33o10	passe	0	0	1	0	0	0	0	0	0	0	0
33o10/46o9	passe	0	0	0	1	0	0	0	0	0	0	0
46o9/32o9	conducao	0	0	0	0	1	0	0	0	0	0	0
32o9/16o10	passe	0	0	0	0	0	1	0	0	0	0	0
16o10/29o10	conducao	0	0	0	0	0	0	1	0	0	0	0
29o10/44o9	passe	0	0	0	0	0	0	0	1	0	0	0
44o9/44o6	passe	0	0	0	0	0	0	0	0	1	0	0
44o6/gol_le	chute	0	0	0	0	0	0	0	0	0	1	0

Matriz de Saida

		48o9	48o8	34o7	33o10	46o9	32o9	16o10	29o10	44o9	44o6	gol_le
48o9/48o8	passe	0	1	0	0	0	0	0	0	0	0	0
48o8/34o7	passe	0	0	1	0	0	0	0	0	0	0	0
34o7/33o10	passe	0	0	0	1	0	0	0	0	0	0	0
33o10/46o9	passe	0	0	0	0	1	0	0	0	0	0	0
46o9/32o9	conducao	0	0	0	0	0	1	0	0	0	0	0
32o9/16o10	passe	0	0	0	0	0	0	1	0	0	0	0
16o10/29o10	conducao	0	0	0	0	0	0	0	1	0	0	0
29o10/44o9	passe	0	0	0	0	0	0	0	0	1	0	0
44o9/44o6	passe	0	0	0	0	0	0	0	0	0	1	0
44o6/gol_le	chute	0	0	0	0	0	0	0	0	0	0	1

Figura 15 – Representação matricial de uma RdP que representa um modelo comportamental de um determinado oponente

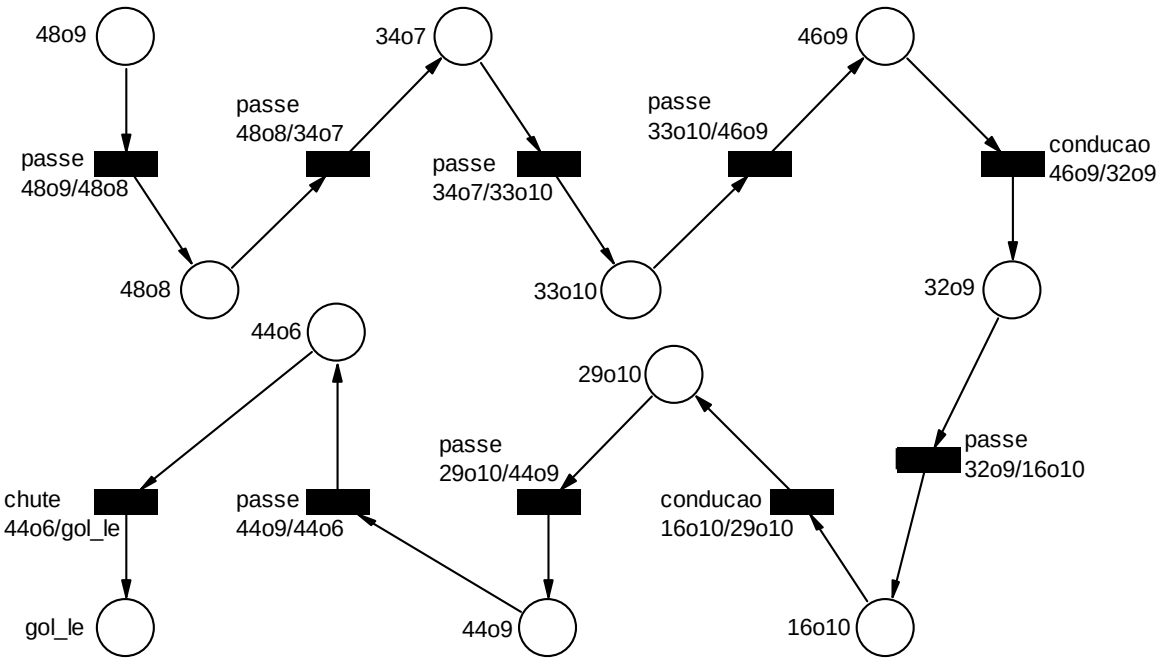


Figura 16 – Representação gráfica de uma RdP que representa um modelo comportamental de um determinado oponente

Um esclarecimento faz-se necessário com relação à forma com que se diferencia a caracterização de um chute e de um passe. O método considera uma transição do tipo chute sempre que uma jogada do oponente está em andamento e o Classificador identifica o estado “Gol do oponente” ou o estado “Cobrança de tiro de meta”, ou seja, sempre que o oponente efetua um gol ou é o responsável por uma saída de bola no fundo do campo. Já uma transição do tipo passe é caracterizada sempre que uma jogada do oponente está em andamento e o Classificador identifica o estado “Troca de detentor de posse de bola” ou “Ocorrência de lateral”, ou seja, sempre que o oponente efetua um passe ou é o responsável por uma saída de bola na lateral do campo.

4.3.2. Conceito de Desvio

Em virtude de limitações impostas pelo *soccerserver*, como, por exemplo, *buffer* limitado para armazenamento de mensagens, conexão UDP e outros fatores, em uma partida que possui duração de 10 minutos, onde ocorrem atualizações do estado do jogo a cada 100ms, ou seja, ocorrem 6000 atualizações no decorrer da partida, são disponibilizadas, em média, apenas 50% das visões globais que compõem a partida. Logo, em virtude da fragmentação mencionada, no recebimento da seqüência de visões globais que constituem a partida, da influência do vento na trajetória da bola e eventuais desvios na trajetória da bola efetuados por jogadores do outro time, foi implementado, no módulo principal, o conceito de desvio.

Um desvio possui a seguinte definição: em um confronto entre um time A e um time B (time oponente), ao ser detectado apenas um toque na bola efetuado por um jogador do time A no meio de uma jogada do time B, este toque é caracterizado como desvio passando a ser desprezado na representação de tal jogada do time oponente. Uma vez que o toque na bola pode realmente não ter ocorrido e ser fruto de um erro de análise ocasionado pela ausência de continuidade na seqüência de visões globais fornecidas, por influência do vento ou presença de ruído na visão global transmitida.

Desta forma, o início de uma jogada do oponente, deixa de ser caracterizado pela classificação do estado “Saída de bola em posse do oponente” ou “Aquisição da

posse de bola pelo time oponente” e passa a ser caracterizado pela identificação do estado “Saída de bola em posse do oponente” ou pela identificação em seqüência dos estados:

- “Aquisição de posse de bola pelo time oponente” e “Gol do oponente”;
- “Aquisição de posse de bola pelo time oponente”, “Estado de posse de bola indefinido” e “Gol do oponente”;
- “Aquisição de posse de bola pelo time oponente”, “Estado de posse de bola indefinido” e “Permanência de posse de bola com jogador oponente”;
- “Aquisição de posse de bola pelo time oponente”, “Estado de posse de bola indefinido” e “Troca de detentor de posse de bola”.

Com relação à caracterização do termino de uma jogada do oponente, esta se dá em virtude da classificação das seguintes seqüências de estados do SMA:

- “Desarme do oponente” e “Fim do primeiro tempo”;
- “Desarme do oponente” e “Gol do time”;
- “Desarme do oponente”, “Estado de posse de bola indefinido” e “Permanência de posse de bola com o time”;
- “Desarme do oponente”, “Estado de posse de bola indefinido” e “Fim do primeiro tempo”;
- “Desarme do oponente”, “Estado de posse de bola indefinido” e “Gol do time”.

4.3.3. Módulo Classificador

Detalhar-se-á agora a forma com que o Classificador identifica os estados do SMA, descritos anteriormente, com base em uma visão global que lhe é fornecida como entrada.

Em virtude das características das entradas fornecidas ao método e visando maximizar a utilização das informações presentes nas mesmas foi proposto um classificador baseado em geometria analítica. Neste ponto se faz necessário um esclarecimento com relação a quais informações pertencentes às MDA´s que descrevem visões globais da partida são consideradas efetivamente para as análises. São levadas em consideração as seguintes informações contidas nas MDA´s:

- ciclo,
- posição da bola,
- e posição dos jogadores.

Em virtude da subdivisão do campo em quadrantes, a posição dos objetos no campo (bola e jogadores), é considerada como sendo a extremidade do triângulo, descrita pelo x e y constantes na MDA, o qual representa a área de localização do mesmo e no caso dos jogadores são desprezadas as informações referentes ao ângulo dos corpos e das cabeças dos mesmos.

Deve-se salientar que a identificação do estado do SMA, nem sempre é possível através da analise apenas das informações contidas na visão global que descreve o instante em que se deseja classificar em qual estado o sistema encontra-se. Na maioria das vezes a analise é feita com base no histórico do SMA, ou seja, levando-se em consideração o comportamento do sistema descrito por uma seqüência de visões globais consecutivas. Para uma melhor compreensão do processo de classificação, serão detalhadas a seguir as condições que levam à identificação de cada um dos estados mencionados anteriormente:

- O estado “Pré-inicial” é caracterizado quando, o valor do ciclo constante na MDA que representa a visão global é igual a 0 (zero);
- O “Estado de posse de bola indefinido” é caracterizado quando, identifica-se que a trajetória da bola teve uma alteração em sua direção e/ou sentido. Verificação esta que é feita com base nas últimas posições que a bola esteve

nos dois ciclos antecessores que tiveram suas visões globais fornecidas pelo servidor, no caso particular onde ainda não foram fornecidas duas visões globais antecessoras à atual, é considerado que a bola mantém mesma direção e sentido. Sempre é considerada uma reta como direção para a trajetória da bola;

- O estado “Saída de bola em posse do oponente” é caracterizado quando, após a ocorrência de um gol, de uma saída de bola na lateral ou fundo do campo, do início da partida ou do segundo tempo da mesma a bola volta ao jogo em posse do time oponente. Situação esta que não é de difícil determinação, uma vez que, após o Classificador ter identificado um dos estados “Pré-inicial”, “Fim do primeiro tempo”, “Gol do oponente”, “Gol do time”, “Ocorrência de lateral” ou “Cobrança de tiro de meta”, basta verificar que jogador será responsável pela saída de bola e identificar se este pertence ao time oponente;

- O estado “Desarme do oponente” é caracterizado quando, após ter sido classificado o “Estado de posse de bola indefinido”, ou seja, após ter se verificado uma mudança na direção e/ou no sentido da trajetória da bola quando esta encontrava-se em posse de um jogador do time oponente, é feita a constatação de que um jogador do outro time é que foi o responsável pelo desvio na trajetória da bola;

- O estado “Troca de detentor de posse de bola” é caracterizado quando, após ter sido classificado o “Estado de posse de bola indefinido”, ou seja, após ter se verificado uma mudança na direção e/ou no sentido da trajetória da bola quando esta encontrava-se em posse de um jogador do time oponente é feita a constatação de que um outro jogador do time oponente é que foi o responsável pelo desvio na trajetória da bola, passando este a ter a posse de bola;

- O estado “Saída de bola em posse do time” é caracterizado quando, após a ocorrência de um gol, de uma saída de bola na lateral ou fundo do campo, do início da partida ou do segundo tempo da mesma a bola não volta ao jogo em

posse do time oponente. Situação esta que é determinada de forma similar à análise feita para a classificação do estado “Saída de bola em posse do oponente”;

- O estado “Fim do primeiro tempo” é caracterizado quando, o valor do ciclo constante na MDA que representa a visão global atual é igual ou superior a 3000 e quando o valor do ciclo constante na MDA que representava a visão global imediatamente anterior a atual fornecida como entrada possuía um valor menor que 3000;

- O estado “Permanência de posse de bola com o time” é caracterizado quando, após ter sido classificado o “Estado de posse de bola indefinido”, ou seja, após ter se verificado uma mudança na direção e/ou no sentido da trajetória da bola quando esta não encontrava-se em posse do time oponente é feita a constatação de que um jogador do próprio time que detinha a posse de bola é que foi o responsável pelo desvio na trajetória da mesma;

- O estado “Permanência de posse de bola com jogador oponente” é caracterizado quando, após ter sido classificado o “Estado de posse de bola indefinido”, ou seja, após ter se verificado uma mudança na direção e/ou no sentido da trajetória da bola quando esta encontrava-se em posse de um jogador do time oponente é feita a constatação de que o próprio jogador que possui a posse da bola é que foi o responsável pelo desvio na trajetória da mesma;

- O estado “Gol do oponente” é caracterizado quando, verifica-se através da visão global fornecida como entrada que a bola encontra-se no quadrante 55 ou 57 e o time oponente encontra-se do lado direito do campo e que na visão global imediatamente anterior a atual fornecida como entrada a bola encontrava-se em um dos quadrantes que compõem a parte interna do campo ou quando, verifica-se através da visão global fornecida como entrada que a bola encontra-se no quadrante 56 ou 58 e o time oponente encontra-se do lado esquerdo do campo e que na visão global imediatamente anterior a

atual fornecida como entrada a bola encontrava-se em um dos quadrantes que compõem a parte interna do campo;

- O estado “Gol do time” é caracterizado quando, verifica-se através da visão global fornecida como entrada que a bola encontra-se no quadrante 55 ou 57 e o time oponente encontra-se do lado esquerdo do campo e que na visão global imediatamente anterior a atual fornecida como entrada a bola encontrava-se em um dos quadrantes que compõem a parte interna do campo ou quando, verifica-se através da visão global fornecida como entrada que a bola encontra-se no quadrante 56 ou 58 e o time oponente encontra-se do lado direito do campo e que na visão global imediatamente anterior a atual fornecida como entrada a bola encontrava-se em um dos quadrantes que compõem a parte interna do campo;

- O estado “Aquisição da posse de bola pelo time oponente” é caracterizado quando, após ter sido classificado o “Estado de posse de bola indefinido”, ou seja, após ter se verificado uma mudança na direção e/ou no sentido da trajetória da bola quando esta não encontrava-se em posse do time oponente é feita a constatação de que um jogador do time oponente é que foi o responsável pelo desvio na trajetória da bola passando este a ter a posse de bola;

- O estado “Ocorrência de lateral” é caracterizado quando, verifica-se através da visão global fornecida como entrada que a bola encontra-se sobre uma das linhas que delimitam os limites laterais do campo e que na visão global imediatamente anterior à fornecida atualmente a bola também encontrava-se sobre a mesma linha. Procedimento este adotado em virtude da bola nunca se encontrar posicionada forra dos limites laterais do campo em partidas de futebol da LSR. É importante salientar que, existe a possibilidade da bola parar sobre uma das linhas que limitam as laterais do campo sem que a bola a tenha a ultrapassado. Porém, a probabilidade disto ocorrer é ínfima, uma vez que, em todos as partidas observadas nenhuma ocorrência deste fato foi

constatada. Logo, não foi detectada a necessidade de sofisticar o processo de identificação da ocorrência de lateral;

- Já o estado “Cobrança de tiro de meta” é caracterizado quando, verifica-se através da visão global fornecida como entrada que a bola encontra-se sobre a posição de cobrança de tiro de meta e que na visão global imediatamente anterior à fornecida atualmente a bola possuía uma distância menor que 2,5 metros da linha que delimita o fundo do campo. Procedimento este adotado em virtude da bola nunca se encontrar posicionada forra dos limites do campo em partidas de futebol da LSR;

- Por fim, o “Estado transitório” é caracterizado quando, após a ocorrência de um gol, da saída da bola dos limites do campo ou do final do primeiro tempo a bola ainda não voltou ao jogo.

4.3.4. Identificação da mudança de sentido e/ou direção na trajetória da bola

Na descrição da forma como são efetuadas as classificações do estado do SMA algumas vezes foi feito menção sobre a caracterização da mudança de sentido e/ou direção na trajetória da bola, será detalhado agora de que forma isto ocorre. Como, foi explanado anteriormente, a posição da bola é descrita por uma área onde esta se localiza, ou melhor, por um retângulo. Também foi exposto que é feita uma aproximação no que refere-se à localização de objetos. Em virtude da subdivisão do campo em quadrantes, é considerada apenas a extremidade do retângulo fornecida como sendo a localização do objeto, no caso a bola. Desta forma, surge o seguinte problema: a figura 17 mostra três retângulos em um plano e evidência que pontos que pertencem a estes retângulos também pertencem a uma mesma reta. No entanto, se considerarmos as extremidades destes retângulos o mesmo não ocorre, como é demonstrado na figura 18. Também foi mencionado que a trajetória da bola é definida por uma reta, sendo assim, percebe-se o problema criado pela aproximação feita na descrição, contida na visão global, para localização de objetos no ambiente, mais especificamente na descrição da localização da bola.

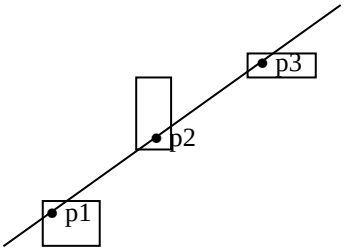


Figura 17 – Distribuição de três retângulos em um plano, evidenciado a formação de uma reta com pontos pertencentes aos mesmos

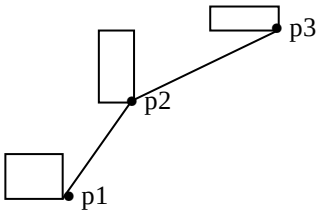


Figura 18 – Distribuição de três retângulos em um plano, evidenciado a não formação de uma reta com pontos pertencentes às extremidades dos mesmos

Buscando minimizar o problema observado, foi feita uma analise em três mil e trinta e nove visões globais, as quais representavam uma determinada partida, coletando as dimensões dos retângulos que representavam as áreas de localização da bola no campo e observando também qual a distância média entre dois centros de retângulos consecutivos, os quais representavam áreas de localização da bola. Com base nas informações coletadas, foi obtido um retângulo com dimensões médias de 0,766908m x 0,645958m e distância média entre dois centros de retângulos consecutivos igual a 2,720181. Com base nestes resultados, estipulou-se que três pontos, que descrevem a trajetória da bola, pertencem à mesma reta (ou descrevem a mesma trajetória) se o ângulo formado pelas retas, obtidas conforme apresentado na figura 19, for menor que 5,143 graus, ou seja, se a tangente deste ângulo for menor que 0,09. É importante salientar que esta abordagem não elimina o problema, e minimiza a probabilidade de ocorrência do mesmo. Caso, em virtude do problema aludido, ocorra uma falha na identificação da mudança de sentido e/ou direção na

trajetória da bola, esta poderá ser eliminada através da caracterização de um desvio, conforme mencionado.

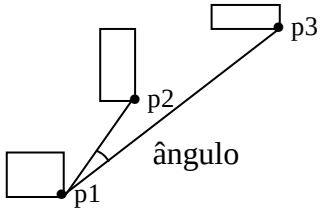


Figura 19 – Distribuição de três retângulos em um plano, evidenciado a formação de ângulo entre retas definidas com pontos pertencentes às extremidades dos mesmos

4.3.5. Identificação do jogador responsável pela mudança de sentido e/ou direção na trajetória da bola

Contudo, o núcleo principal do Classificador é a parte que identifica após a caracterização da mudança de sentido e/ou direção na trajetória da bola, que jogador foi responsável por tal alteração. Para o detalhamento de como ocorre este processo partiremos da observação da seqüência de quadros apresentada na figura 20. O primeiro quadro retrata o jogador número 11 do time oponente com a posse de bola e efetuando um chute na mesma. No segundo quadro observa-se o deslocamento da bola, que no terceiro quadro tem sua trajetória alterada pela ação do jogador número 4 do outro time. Já os quadros quatro e cinco apresentam o deslocamento da bola em sua nova trajetória.

No entanto, como foi mencionado anteriormente, o servidor não fornece a seqüência completa de visões globais que compõem a partida. Logo, em uma analise da seqüência de quadros apresentada na figura 21, a qual omite o terceiro quadro, chegam-se às seguintes conclusões: no primeiro quadro tem-se o jogador número 11 do time oponente com a posse de bola e efetuando um chute na mesma; No segundo quadro observa-se o deslocamento da bola, que no quarto quadro tem uma trajetória divergente à apresentada nos quadros anteriores. No entanto, como identificar qual jogador foi o responsável pela mudança na trajetória da bola?

Considerar o jogador mais próximo da bola no quadro que antecede a identificação da mudança de trajetória, ou seja, responsabilizar o jogador número 7 do time oponente ou considerar o jogador mais próximo da bola no quadro que ocorre a identificação da mudança de trajetória, ou seja, responsabilizar o jogador número 8 do outro time pela mudança na trajetória da bola. A questão é que, como verificamos na seqüência completa das visões globais, apresentada na figura 20, o jogador responsável pela mudança na trajetória da bola foi o jogador número quatro do outro time.

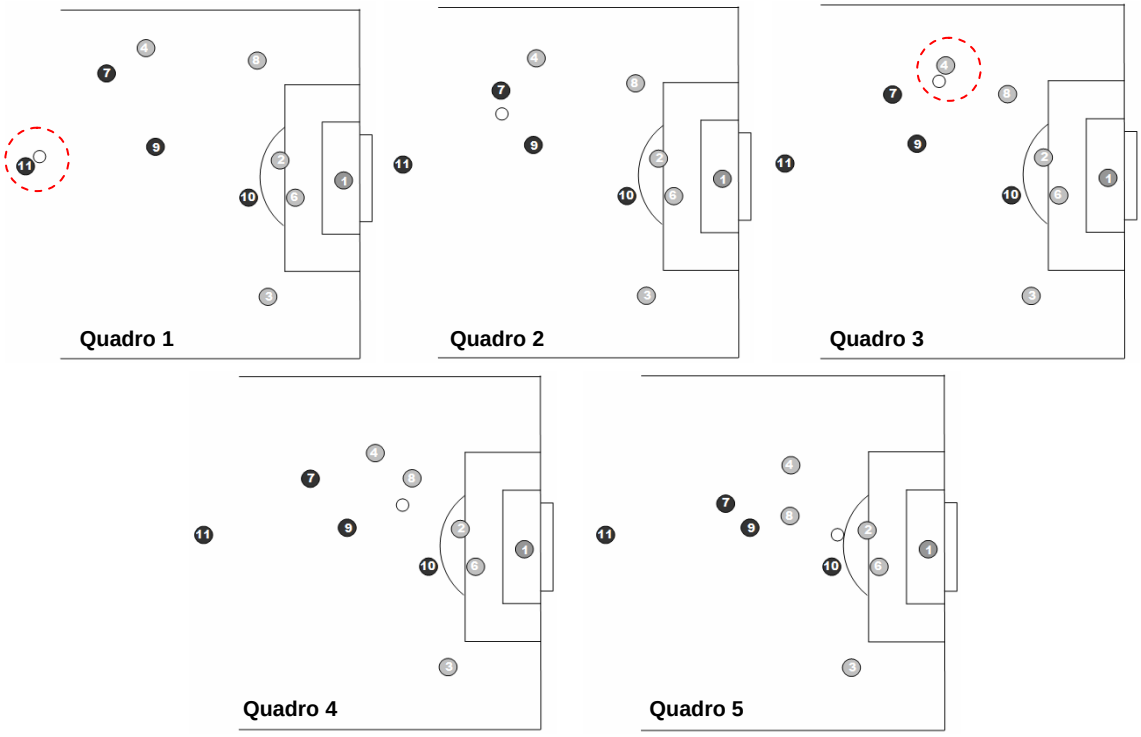


Figura 20 – Seqüência de quadros de uma determinada partida

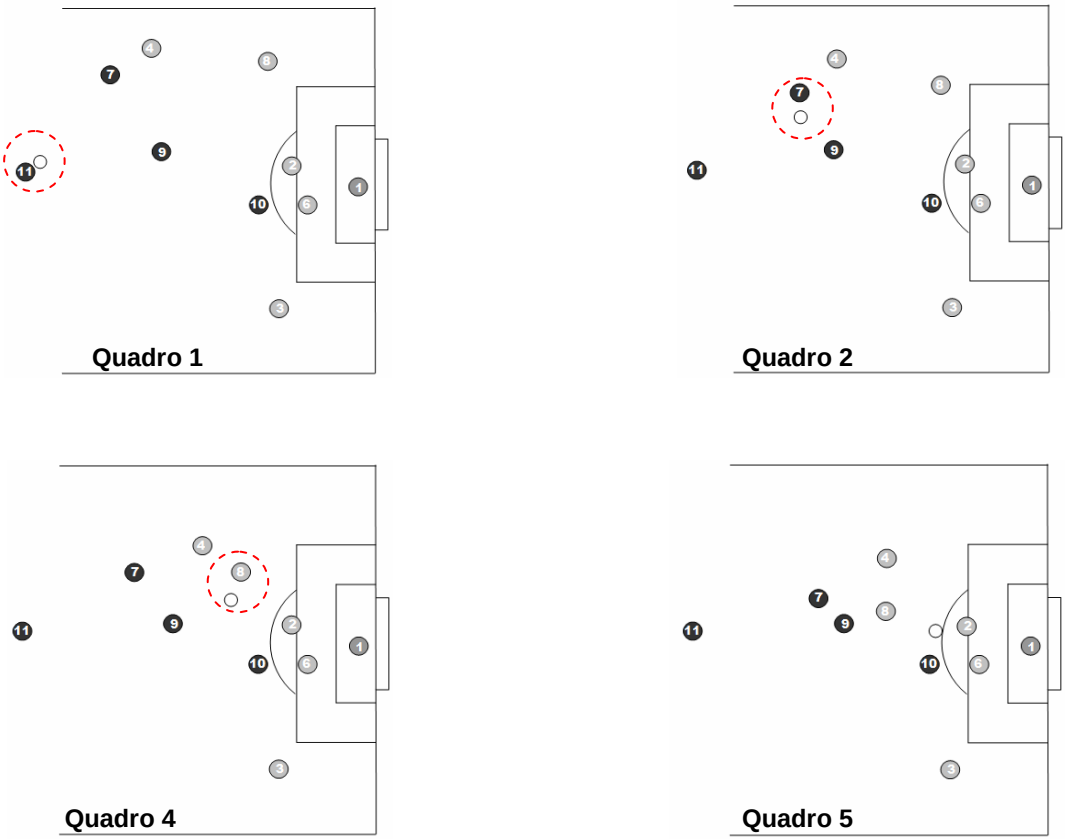


Figura 21 – Seqüência de quadros, fragmentada, de uma determinada partida

Para sanar o problema exposto, propôs-se o seguinte procedimento: ao se identificar a mudança na trajetória da bola, o que ocorre na situação em análise no quarto quadro (evidenciado na figura 22), observa-se a posição da mesma no próximo quadro disponibilizado pelo servidor, traçando-se uma nova trajetória com base na posição da bola no quadro que foi identificada a mudança em sua trajetória e a posição da mesma no quadro que o sucede, localizando o ponto de intersecção entre as duas trajetórias, conforme apresentado na figura 23.

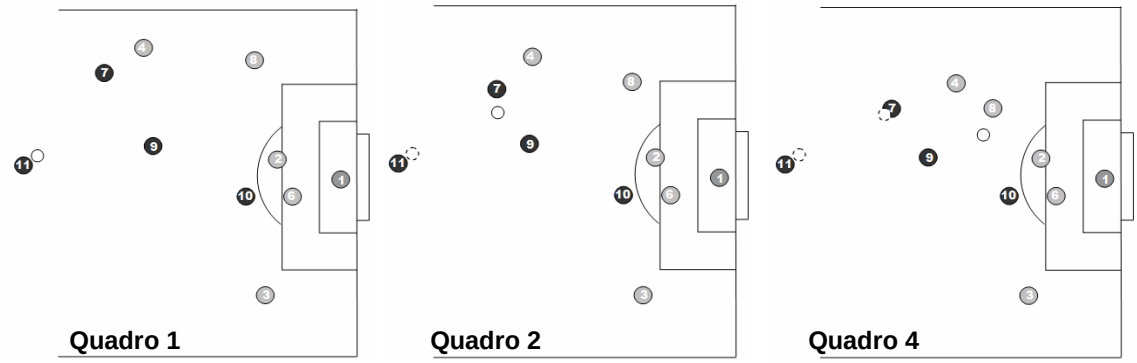


Figura 22 – Seqüência de quadros de uma determinada partida, considerando o rastro da bola

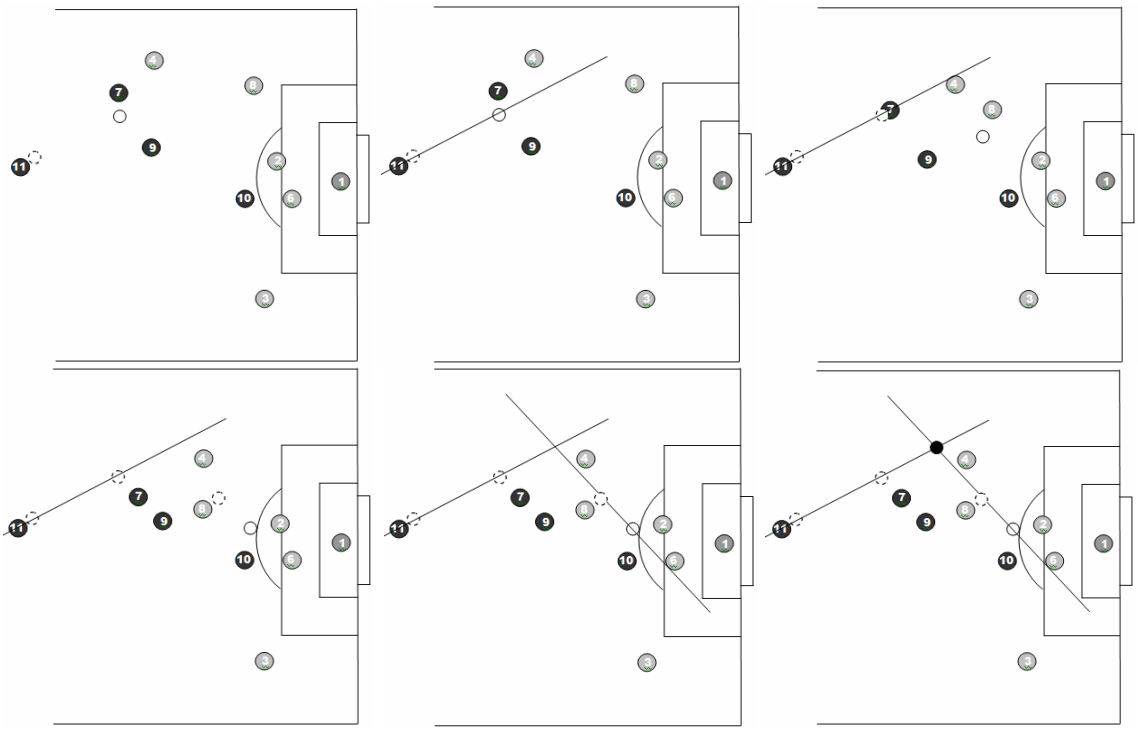


Figura 23 – Seqüência de quadros de uma determinada partida, localização do provável ponto de mudança de trajetória

Em posse do ponto de intersecção entre as trajetórias defini-se uma área de análise, delimitada por uma circunferência que tem como centro o ponto de intersecção entre as trajetórias e raio igual a 2m acrescidos da distância do centro da circunferência ao ponto que descreve a posição da bola no quadro que antecede o quadro que evidencia a mudança na trajetória da bola. Isto garante a presença na região de análise de pelo menos um jogador uma vez que mesmo no caso em que o quadro que antecede o quadro que evidencia a mudança na trajetória da bola venha a descrever o instante da ocorrência de tal mudança, já que neste caso o raio mínimo, da área de análise, será 2m o que garantirá a presença ao menos do jogador responsável pelo desvio na trajetória da bola, uma vez que este obrigatoriamente possui raio menor que 2m. Tendo-se especificado a área de análise, esta ocorre da seguinte forma: verifica-se no quadro que evidencia a mudança na trajetória da bola e no quadro que o antecede quais jogadores possuem sua localização dentro da área de analise em pelo menos um dos quadros. De posse desta informação traça-se uma reta para cada jogador que encontrava-se na área de analise utilizando suas posições nos quadros mencionados. Após, verifica-se qual a menor distância entre o centro da área de analise e as retas traçadas. Sendo responsabilizado pelo desvio o

jogador que gerou a reta mais próxima do centro da área de analise. Para uma melhor compreensão do processo é apresentada, na figura 24, a aplicação do mesmo sobre a situação que estava sendo discutida anteriormente. Observa-se que mesmo sem a presença do terceiro quadro, utilizando-se do procedimento proposto, conclui-se que o jogador número 4 do outro time é que foi o responsável pela mudança na trajetória da bola.

Uma observação se faz necessária, no caso de se evidenciar mais de uma reta traçada com a mesma distância ao centro da área de análise e esta distância corresponder à menor distância constatada entre as retas traçadas e o centro da área de análise, o jogador com menor número, responsável pela definição dos pontos que traçaram uma das retas de conflito, será responsabilizado pelo desvio.

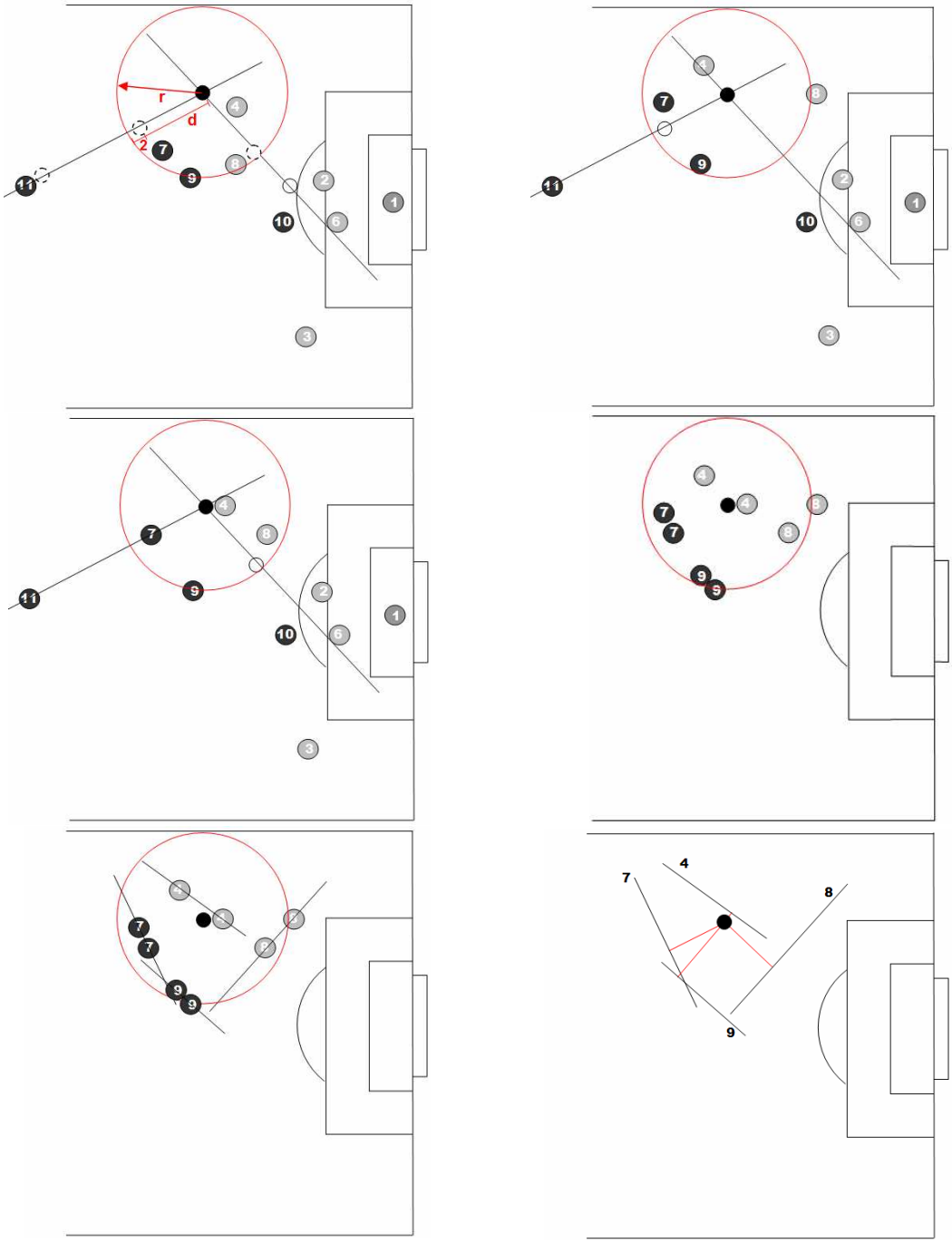


Figura 24 – Processo de identificação do jogador responsável pela mudança na trajetória da bola

5. IMPLEMENTAÇÃO E ESTUDOS DE CASO

Com a finalidade de contrapor modelos comportamentais obtidos pelo MOMCoTO com os respectivos comportamentos evidenciados através da observação dos *logfiles* dos jogos ocorridos, efetuou-se a implementação do mesmo. Considerando que o método foi idealizado com uma estrutura em módulos e objetivando uma melhor manutenibilidade e reusabilidade do sistema a ser implementado, foi feita a opção pelo paradigma de programação orientado a objetos, escolhendo-se a linguagem C++ para levar à prática o projeto. Esta decisão foi alicerçada na possibilidade de se ter métodos implementando os módulos do MOMCoTO.

O detalhamento de cada uma das classes definidas assim como o respectivo código das mesmas encontram-se no Apêndice II.

5.1. Estudos de caso

Para efetuar uma análise qualitativa do MOMCoTO torna-se necessário anexar a implementação do mesmo ao código de um time de futebol da LSR, mais especificamente ao trecho de código que implementa o treinador *on line* do time em questão. Levando-se em consideração a complexidade proveniente da implementação de um time e visando minimizar a dedicação de esforços a atividades meio, no que diz respeito à modelagem comportamental do time oponente, foi feita a opção pela utilização do código fonte base, disponibilizado, do time UvA Trilearn 2003. Apesar deste possuir apenas agentes com funções básicas implementadas, o mesmo disponibiliza a estrutura necessária para a implementação do MOMCoTO.

No trecho de código que implementa o treinador *on line* do time base UvA Trilearn 2003, mais especificamente no arquivo BasicCoach.cpp, no método void BasicCoach::mainLoopNormal( ), foi inserido o seguinte trecho de código:

```
FILE *pf=fopen("visoes_globais.txt","w");
int atual,antigo=-1;
while (1)
{
   char str[MAX_MSG];
   C->receiveMessage( str, MAX_MSG );
```

```
    if (encontrou(str, "see_global"))
    {
          sscanf(str,"(see_global %d (",&atual);
          if(atual>antigo)
          {
             antigo=atual;
             fprintf(pf,"%s\n",str);
          }
    }
}
fclose(pf);
```

A função encontrou(), também inserida, recebe duas *strings* e retorna verdadeiro se a segunda *string* recebida como entrada esta contida na primeira *string*, retornando falso caso contrário. Com estas alterações no código, ao ser efetuada uma partida entre o time em questão e um adversário, será gerado ao final da mesma, o arquivo texto “visoes_globais.txt”, o qual conterá todas as visões globais da partida disponibilizadas pelo servidor.

Dando continuidade ao propósito de efetuar uma análise qualitativa do modelo comportamental do time oponente em uma partida de futebol entre equipes de robôs autônomos, foi escrito o seguinte código:

```
#include "MOMCoTO.h"
main()
{
  RdP modelo_do_oponente;
  FILE *pf, *pf2, *pf3;
  Construtor_RdP c("\"Oxente_2005_V1_2\"");
  char visao_global[2000];
  if (!(pf=fopen("visoes_globais.txt","r")))
    exit (1);
  if (!(pf2=fopen("RdP_modelo_comportamental.txt","w")))
    exit (1);
  if (!(pf3=fopen("Frequencias_de_ocorrencia_das_transicoes_da_RdP.txt", "w")))
    exit (1);
  while (!feof(pf))
  {
    fgets(visao_global,2000,pf);
    fscanf(pf,"\n");
    c.atualizar_RdP(modelo_do_oponente, visao_global);
  }
  modelo_do_oponente.escrever_RdP_em_arquivo(pf2);
  modelo_do_oponente.escrever_RdP_com_frequencia_em_arquivo(pf3);
```

```
    fclose(pf3);
    fclose(pf2);
    fclose(pf);
}
```

o qual se utiliza do MOMCoTO aplicando-o sobre o arquivo “visoes_globais.txt”, que contém as visões globais de uma determinada partida, e gerando dois arquivos como saída. O arquivo denominado “RdP_modelo_comportamental.txt” contendo as matrizes de entrada e saída que descrevem a RdP que representa o modelo comportamental do oponente. E o arquivo “Frequencias_de_ocorrencia_das_ transicoes_da_RdP.txt” contendo as freqüências de ocorrência das transições da RdP.

O processo de avaliação do modelo comportamental gerado ocorre da seguinte forma: quando uma determinada partida é executada para geração do arquivo contendo as visões globais, a mesma, também é gravada através da geração dos arquivos de *log*. Em seguida o arquivo de *log* da partida é executado possibilitando uma observação da mesma ciclo a ciclo, permitindo, também, retroceder e re-observar, quantas vezes se queira, um determinado evento. Desta forma, são anotadas as informações referentes à primeira jogada do oponente, à ocorrência de gols, bolas fora nas laterais e chutes a gol sem êxito que ocasionaram cobranças de tiros de meta. Em seguida o modelo comportamental do time oponente é gerado, através do MOMCoTO, utilizando-se o código descrito anteriormente, e o mesmo é comparado com as informações obtidas através da observação do *logfile*.

Para uma avaliação mais completa da eficácia do método proposto, a análise do modelo comportamental gerado foi aplicada sobre dez partidas realizadas com dois oponentes distintos. Cinco foram realizadas com o time FCPortugal2002 participante da RoboCup do ano de 2002 e as outras cinco com o time UvATrilearn2003 campeão da RoboCup do ano de 2003. Sendo que, cinco dessas partidas foram efetuadas com o oponente do lado esquerdo do campo e outras cinco com o oponente do lado direito do campo. As tabelas a seguir apresentam os resultados das análises efetuadas sobre os modelos comportamentais gerados em cada uma das partidas realizadas. Os arquivos contendo as visões globais das partidas analisadas, que foram entradas para o MOMCoTO, assim como os modelos

comportamentais gerados na análise da partidas pelo método, ou melhor, as saídas do MOMCoTO, encontram-se disponíveis no endereço “www.univasf.edu.br/~marcelo.linder/arquivos_MOMCoTO”.

Para uma melhor compreensão das informações constantes nas tabelas seguintes, deve-se ter em mente que assim como em jogos reais na Liga Simulada da RoboCup ocorrem chutes prensados e às vezes a bola passa através de jogadores sem que estes toquem na bola, simulando, por exemplo, a situação em que a bola passa por baixo da perna de um jogador em um drible, em situações como estas ocorrem equívocos nas análises que geram o modelo comportamental do oponente. Além disso, mais uma observação se faz necessária, devido aos problemas relatados anteriormente, como, por exemplo, fragmentação da seqüência de visões globais fornecidas pelo servidor e ação do vento sobre a bola, também são fatores geradores de equívocos nas análises que geram o modelo, como conseqüência, por exemplo, uma finalização prematura de uma jogada pode ser caracterizada. Situação esta que se evidencia na caracterização da primeira jogada efetuada pelo oponente no terceiro jogo analisado, a qual teve seu final caracterizado prematuramente, fato este que justifica os baixos percentuais de acerto apresentados na tabela referentes aos passes e conduções apresentados na primeira jogada da partida em questão.

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Primeira partida UvA Trilearn (oponente do lado direito do campo)	Gol	37	37	37	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	37	37	29	78,38%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	37	37	37	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	5	4	3	60%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	5	4	4	80%
	Bola ultrapassou os limites laterais do campo	5	5	5	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	5	5	3	60%
	Cobrança de lateral (observado o time que cobrou o lateral)	5	5	5	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	7	7	7	100%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	2	2	2	100%
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 2 – Análise da primeira partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Segunda partida UvA Trilearn (oponente do lado esquerdo do campo)	Gol	43	43	43	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	43	43	31	72,09%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	43	43	43	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	4	4	3	75%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	4	4	4	100%
	Bola ultrapassou os limites laterais do campo	3	3	3	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	3	3	2	66,6%
	Cobrança de lateral (observado o time que cobrou o lateral)	3	3	3	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	5	3	3	60%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	1	1	1	100%
	Finalização da primeira jogada do oponente	1	1	0	0%

Tabela 3 – Análise da segunda partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Terceira partida UvA Trilearn (oponente do lado direito do campo)	Gol	34	34	34	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	34	34	28	82,35%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	34	34	34	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	2	1	1	50%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	2	1	1	50%
	Bola ultrapassou os limites laterais do campo	5	4	4	80%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	5	4	3	60%
	Cobrança de lateral (observado o time que cobrou o lateral)	5	4	4	80%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	4	1	1	25%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	3	1	1	33,33%
	Finalização da primeira jogada do oponente	1	1	0	0%

Tabela 4 – Análise da terceira partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Quarta partida UvA Trilearn (oponente do lado esquerdo do campo)	Gol	34	34	34	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	34	34	31	91,17%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	34	34	34	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	5	3	2	40%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	5	3	3	60%
	Bola ultrapassou os limites laterais do campo	2	2	2	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	2	2	2	100%
	Cobrança de lateral (observado o time que cobrou o lateral)	2	2	2	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	5	4	4	80%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	1	1	1	100%
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 5 – Análise da quarta partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Quinta partida UvA Trilearn (oponente do lado direito do campo)	Gol	40	40	40	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	40	40	30	75%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	40	40	40	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	3	1	1	33,33%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	3	1	1	33,33%
	Bola ultrapassou os limites laterais do campo	4	4	4	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	4	4	2	50%
	Cobrança de lateral (observado o time que cobrou o lateral)	4	4	4	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	1	1	1	100%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	0	0	0	-
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 6 – Análise da quinta partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Sexta partida FC Portugal (oponente do lado esquerdo do campo)	Gol	6	6	6	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	6	6	6	100%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	6	6	6	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	0	0	0	-
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	0	0	0	-
	Bola ultrapassou os limites laterais do campo	19	18	18	94,73%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	19	18	13	68,42%
	Cobrança de lateral (observado o time que cobrou o lateral)	19	18	18	94,73%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	0	0	0	-
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	1	1	1	100%
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 7 – Análise da sexta partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Sétima partida FC Portugal (oponente do lado direito do campo)	Gol	13	13	13	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	13	13	9	69,23%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	13	12	12	92,30%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	1	0	0	0%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	1	0	0	0%
	Bola ultrapassou os limites laterais do campo	10	10	10	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	10	10	5	50%
	Cobrança de lateral (observado o time que cobrou o lateral)	10	10	10	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	2	2	2	100%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	1	1	1	100%
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 8 – Análise da primeira partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Oitava partida FC Portugal (oponente do lado esquerdo do campo)	Gol	7	7	7	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	7	7	5	71,42%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	7	7	7	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	6	4	3	50%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	6	4	4	66,66%
	Bola ultrapassou os limites laterais do campo	15	15	15	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	15	15	11	73,33%
	Cobrança de lateral (observado o time que cobrou o lateral)	15	15	15	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	1	1	1	100%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	0	0	0	-
	Finalização da primeira jogada do oponente	1	1	0	0%

Tabela 9 – Análise da oitava partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Nona partida FC Portugal (oponente do lado direito do campo)	Gol	6	6	6	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	6	6	1	16,66%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	6	6	6	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	2	2	2	100%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	2	2	2	100%
	Bola ultrapassou os limites laterais do campo	12	12	12	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	12	12	10	83,33%
	Cobrança de lateral (observado o time que cobrou o lateral)	12	12	12	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	2	2	2	100%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	0	0	0	-
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 10 – Análise da nona partida

Jogo	Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Décima partida FC Portugal (oponente do lado esquerdo do campo)	Gol	8	8	8	100%
	Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	8	8	5	62,5%
	Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	8	8	8	100%
	Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	5	4	4	100%
	Cobrança de tiro de meta (observado o time que efetuou a cobrança)	5	4	4	100%
	Bola ultrapassou os limites laterais do campo	13	13	13	100%
	Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	13	13	10	76,92%
	Cobrança de lateral (observado o time que cobrou o lateral)	13	13	13	100%
	Final do primeiro tempo	1	1	1	100%
	Início do segundo tempo (observado que time saiu com posse de bola)	1	1	1	100%
	Início da primeira jogada do oponente	1	1	1	100%
	Passes ocorridos na primeira jogada do oponente (observado de que jogador em que quadrante para qual jogador em qual quadrante)	1	1	1	100%
	Conduções ocorridas na primeira jogada do oponente (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	0	0	0	-
	Finalização da primeira jogada do oponente	1	1	1	100%

Tabela 11 – Análise da décima partida

Como pode ser observado, foi analisado o quão fiel foi o modelo gerado com relação à primeira jogada executada pelo oponente, em cada um dos jogos, uma vez que comparar o modelo gerado a cada uma das jogadas efetuadas pelo oponente no decorrer dos jogos torna-se inviável. Como foi mencionado anteriormente, as

jogadas podem ser finalizadas através da saída de bola dos limites do campo, da realização de um gol, do final do primeiro tempo ou da ocorrência de um desarme, observou-se o quão fiel foi o modelo com relação à caracterização de bolas fora e do final do primeiro tempo, observando-se também mais alguns detalhes evidenciados nas tabelas.

Para finalizar as análises, a seguir consta uma tabela com uma análise geral dos modelos gerados nas dez partidas examinadas.

Evento observado	N° de ocorrências	N° de ocorrências identificadas com imprecisão	N° de ocorrências identificadas corretamente	Percentual de acerto
Gol	228	228	228	100%
Chute a gol com êxito do oponente (observado o quadrante onde ocorreu o chute e o jogador que efetuou o chute)	228	228	175	76,75%
Saída de bola após ocorrência de gol (observado o time que efetuou a saída de bola)	228	227	227	99,56%
Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)	33	23	19	57,57%
Cobrança de tiro de meta (observado o time que efetuou a cobrança)	33	23	23	69,69%
Bola ultrapassou os limites laterais do campo	88	86	86	97,72%
Identificação do time responsável pela bola ter ultrapassado os limites laterais do campo	88	86	61	69,31%
Cobrança de lateral (observado o time que cobrou o lateral)	88	86	86	97,72%
Final do primeiro tempo	10	10	10	100%
Início do segundo tempo (observado que time saiu com posse de bola)	10	10	10	100%
Início das primeiras jogadas dos oponentes	10	10	10	100%
Passes ocorridos nas primeiras jogadas dos oponentes (observado de que jogador em que quadrante para qual jogador em qual quadrante)	28	22	22	78,57%
Conduções ocorridas nas primeiras jogadas dos oponentes (observado de que quadrante para qual quadrante ocorreu a condução e qual jogador foi responsável pela mesma)	9	7	7	77,77%
Finalizações das primeiras jogadas dos oponentes	10	10	7	70%

Tabela 12 – Análise das partidas

Certos comentários se fazem necessários com relação a alguns percentuais de acertos contidos na tabela 12:

- Em uma primeira análise, talvez se associe o fato da obtenção de 100% no percentual de acerto na identificação dos eventos “Gol”, “Final do primeiro tempo” e “Início do segundo tempo (observando que time saiu com posse de bola)” com o fato do servidor de simulação fornecer estas informações. No entanto, deve-se lembrar que nenhuma informação adicional às visões globais são utilizadas na geração do modelo;

- Outra observação relevante é que além dos fatores, mencionados anteriormente, que interferem no acerto na modelagem como, por exemplo, os que influenciaram na criação do conceito de desvio, a identificação de alguns eventos dependem de informações contidas em determinadas visões globais e como são fornecidas apenas, em média, 50% das visões globais que compõem a partida em determinados casos a ausência de uma visão global especifica pode vir a tornar impossível a identificação de determinados eventos. Um exemplo, bem expressivo, do que foi dito encontra-se nos baixos percentuais de acerto na identificação do evento “Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)”. Devemos nos lembrar que a identificação da ocorrência do evento “Chute a gol sem êxito do oponente, onde a bola saiu no fundo do campo (observado o time que efetuou o chute)” depende da observação do posicionamento preciso da bola sobre o ponto de cobrança de tiro de meta, como em determinadas cobranças a bola permanece sobre esta posição apenas por um ciclo, conseqüentemente sem o recebimento da visão global referente ao mesmo é impossível, através do método proposto, identificar a ocorrência do evento.

Cabem aqui colocações referentes ao consumo de memória e tempo necessário para modelar o comportamento do time oponente. Em seu pico máximo de consumo de memória, identificado com base nos estudos de caso efetuados, pico este que ocorre quando todas as visões globais referentes a uma determinada partida foram

analisadas e o modelo comportamental completo do time oponente encontra-se na memória, o consumo médio é de 1.288 Kbytes. Sendo que desta quantidade de memória apenas 144 Kbytes são utilizados para armazenar o modelo comportamental. Este baixo consumo de memória se justifica pela abordagem adotada para resolução do problema e pela estrutura de dados que se utilizou para armazenar a RdP, a qual, além de consumir pouca memória, possibilita um acesso muito rápido aos dados armazenados. As matrizes de entrada e saída que descreve a RdP foram armazenada em matrizes esparsas através do conceito de listas cruzadas alocadas dinamicamente.

Além do baixo consumo de memória necessário para o armazenamento do modelo comportamental, também merece um especial destaque o tempo necessário para que todas as visões globais referentes a uma partida sejam analisadas e o respectivo modelo do oponente seja gerado. Como foi exposto anteriormente, uma partida é comporta por 6000 visões globais, porém, são disponibilizadas para análise, em média, 50% destas visões. Ao efetuarem-se os estudos de caso evidenciou-se que o tempo necessário para gerar o modelo comportamental do oponente é, em média, 1,3818 segundos. Ou seja, uma visão global é analisada e suas informações relevantes inseridas na rede em aproximadamente 0,461ms. Considerando que cada visão global é gerada a cada 100ms, ou seja, o intervalo entre visões globais consecutivas é de no mínimo 100ms, o tempo de análise do método pode ser considerado desprezível. Outro dado importante é o fato dos testes mencionados anteriormente terem sido efetuados em uma máquina com as seguintes configurações: processador Mobile AMD Semprom de 787 MHz e 256 MB de RAM. Estas informações demonstram que o MOMCoTO pode ser aplicado no decorrer de uma partida sem ocasionar uma perda computacional relevante, uma vez que suas análises demandam custo computacional desprezível. Desta forma o método pode ser utilizado, por uma equipe no decorrer de uma partida, para gerar um modelo parcial do comportamento do time oponente (modelo este atualizado a cada nova visão global recebida), utilizando-o para conjecturar a respeito de movimentos futuros do adversário, com o objetivo de torná-los ineficientes ou inviabilizá-los.

6. CONCLUSÕES E TRABALHOS FUTUROS

Esta dissertação apresentou um método para obtenção do modelo comportamental de um Sistema Multiagente. Como ambiente de experimentação foi utilizado o futebol entre equipes de robôs autônomos, sendo alvo da modelagem o comportamento do time oponente. Para fins de validação o método proposto foi implementado e aplicado sobre partidas da LSR, gerando RdP´s que descrevem os modelos comportamentais dos respectivos SMA´s. No entanto, é necessário salientar que o MOMCoTO pode ser aplicado em partidas entre robôs reais, necessitando apenas a utilização de uma câmera posicionada sobre o campo.

6.1. Conclusões

O trabalho desenvolvido teve como premissa não manipular as características do domínio de aplicação no sentido de minimizar ou eliminar suas limitações. Desta forma, o MOMCoTO teve de ser robusto o suficiente para, apesar da complexidade e limitações do domínio no qual seria aplicado, ser capaz de gerar um modelo fiel. Conforme se evidencia por meio dos resultados apresentados na tabela 12, o método se mostrou eficaz, em virtude do mesmo ter obtido um elevado percentual de acerto em suas análises, apesar da existência de restrições impostas pelo domínio de aplicação e considerando a complexidade intrínseca à modelagem comportamental de SMA´s.

Baseado em uma análise do que foi exposto no capítulo anterior, evidencia-se que os objetivos deste trabalho foram atingidos, uma vez que o método proposto gera um modelo comportamental satisfatório, com um tempo de análise desprezível e com um baixo consumo de memória. Estas informações demonstram que o MOMCoTO pode ser aplicado no decorrer de uma partida sem ocasionar uma perda computacional relevante, possibilitando assim uma exploração das informações contidas no modelo comportamental parcial gerado. A referida exploração do modelo comportamental do time oponente pode ocorrer de diversas formas, por exemplo, pode-se utilizá-lo, no decorrer da partida, para identificar tendências comportamentais do oponente, como jogadas ensaiadas, que ao serem identificadas podem ser inviabilizadas.

6.2. Trabalhos Futuros

O trabalho desenvolvido pode servir como base para uma grande gama de outros trabalhos:

- A complementação do método proposto objetivando a sofisticação do modelo comportamental gerado, passando a considerar mais variáveis presentes no ambiente, como, por exemplo, marcos temporais, probabilidades de ocorrência e paralelismo entre ações identificadas. Possibilitando assim, conjecturar sobre movimentos futuros do adversário tendo como base um histórico comportamental mais completo;

- O MOMCoTO pode ser aplicado sobre jogos entre robôs reais, para isto deve ser implementado um módulo para o pré-processamento das imagens do campo e alguns ajustes devem ser feitos com relação à classificação de alguns estados do SMA, como, por exemplo, deve-se adaptar a forma como é classificado o estado de “ocorrência de lateral”;

- A exploração do modelo comportamental gerado: utilizando-o como base para que um time construa uma estratégia eficiente para inviabilizar a execução de um ataque bem sucedido por parte do time oponente. Considerando que, por intermédio do histórico comportamental pode-se, no decorrer da partida, identificar quais agentes são responsáveis pelas finalizações das jogadas; e em quais regiões do campo estas finalizações costumam ocorrer. Pode-se ainda utilizar as informações contidas no modelo comportamental para aprimorar as táticas de ataque de uma determinada equipe, incorporando características positivas encontradas no modelo comportamental do time oponente;

- Além das possibilidades citadas de aprofundamento do trabalho desenvolvido, pode-se avaliar a aplicabilidade do MONCoTO sobre outros domínios.

REFERÊNCIAS BIBLIOGRÁFICAS

AXEBOT. **Projeto Axebot: Agentes Autônomos.** Disponível em: <http://www.axebot.ufba.br>. Acessado em dezembro de 2007.

CARUANA, R.; FREITAG, D., **Greedy Attribute Selection**. In 11th Proceedings of the 11th International Conference on Machine Learning (ICML), 1994.

CHENY, M. *et al.* **RoboCup Soccer Server – for Soccer Server Version 7.07 and later**, 2002.

DESEL, J. **Place/Transitions Nets I**, Introductory Tutorial Petri Nets, Petri Nets 2000, 21st International Conference on Application And Theory of Petri Nets, Denmark, pp 111-160, 2000.

DORAIS, G. *et al*. **Adjustable Autonomy for Human-Centered Autonomous Systems.** Working notes of the Sixteenth International Joint Conference on Artificial Intelligence Workshop on Adjustable Autonomy Systems, 1999.

FINE, S.; SINGER, Y. and TISHBY, N., **Hierarchical Hidden Markov Model: Analysis and applications.** Machine Learning, 32(1), 1998.

GIRAULT, C., VALK, R. **Petri Nets for Systems Engineering: A Guide for Modeling, Verification and Application**, Springer Verlag, 2002.

HUANG, D.; LI, K.; IRWIN G., **Intelligent Control and Automation: International Conference on Intelligent Computing**, p. 668-671, Publisher Springer, 2006.

LINDER, M., COSTA, A. **Modelagem Comportamental de Sistemas Multiagentes**, VI Encontro Nacional de Inteligência Artificial - ENIA/CSBC, 2007, Rio de Janeiro. Anais do XXVII Congresso da Sociedade Brasileira de Computação - CSBC, 2007.

MACIEL, P. R., *et al*. **Introdução às Redes de Petri e Aplicações**, X Escola de Computação, Campinas SP, 1996.

MECATEAM. **Projeto Mecateam: Futebol entre Robôs Autônomos.** Disponível em: <http://twiki.im.ufba.br/bin/view/Mecateam>. Acessado em dezembro de 2007.

MINSKY, M. **A framework to represent knowledge**, In The Psychology of Computer Vision, pages 211–277. McGraw-Hill, 1975.

MURATA, T. **Petri Nets: Properties, Analysis and Applications**, Proceedings of the IEEE, Vol. 77, pp. 541-580, 1989.

NAIR, R.; TAMBE, M. **Automated Assistants for Analyzing Team Behaviors**. 2002 Kluwer Academic Publishers.

PELED, D.; TSAY, Y., **Automated Technology for Verification and Analysis**, Third International Symposium, p. 279, Taipei, Taiwan, October 4-7, 2005.

PENHA, D. O. , *et al*. **Modelagem de Sistemas Computacionais usando Redes de Petri: aplicação em projeto, análise e avaliação.** Anais da IV Escola Regional de Informática RJ/ES, 2004.

PETERSON, J. L. **Petri Nets**, ACM Computing Surveys, vol. 9, No. 3, pp. 223-252, 1977.

PETERSON, J. L. **Petri Net Theory and the Modeling of Systems**, Englewood Cliffs, Prentice Hall, 1981.

RAINES, T.; TAMBE, M.; MARSELLA, M. **Agent Assistants for Team Analysis**. AI Magazine, Vol 21, Num 3, pp 27-31, Fall 2000.

RILEY, P.; VELOSO, M. **Coaching a Simulated Soccer Team by Opponent Model Recognition**, In Proceedings of the Fifth International Conference on Autonomous Agents, 2001.

ROBOCUP. **The RoboCup Federation**. Disponível em: <http://www.robocup.org>. Acessado em dezembro de 2007.

RUSSELL, S.; NORVIG, P., **Artificial Intelligence: A Modern Approach.** Prentice-Hall, Inc., 1995.

SILVA, C. **Modelagem Comportamental para Agentes Autônomos em Ambientes Reais**. 2003. 82 p. – Tese – M.Sc. em Engenharia da Computação – Área de Concentração Geomática – Universidade do Estado do Rio de Janeiro – UERJ, Rio de Janeiro – RJ.

STONE, P.; RILEY, P. , VELOSO M., **The CMUnited-99 Champion Simulator Team**, RoboCup-99: Robot Soccer World Cup III, p.35-48, January 2000.

STONE, P.; VELOSO, M., **Using Decision Tree Confidence Factors for Multiagent Control**. In Proceedings of Internationel Conference on Autonomus Agents, 1998.

SUKTHANKAR, G.; SYCARA, K., **Automatic Recognition of Human Team Behaviors**, Robotics Institute, Carnegie Mellon University, in Modeling Others from Observations (MOO 2005), Workshop at the International Joint Conference on Artificial Intelligence (IJCAI), Edinburgh, Scotland, July 2005.

WEISS, G. **Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence**, MIT Press, 1999.

APÊNDICE I

(see_global 0 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 0 0 0 0) ((p "Oxente_2005_V1_2" 1 goalie) -45 0 0 0 0 0) ((p "Oxente_2005_V1_2" 2) -16 16 0 0 -45 0) ((p "Oxente_2005_V1_2" 3) -17 5 0 0 -16 0) ((p "Oxente_2005_V1_2" 4) -17 -5 0 0 17 0) ((p "Oxente_2005_V1_2" 5) -16 -16 0 0 46 0) ((p "Oxente_2005_V1_2" 6) -8 0 0 0 0 0) ((p "Oxente_2005_V1_2" 7) -5 10 0 0 -63 0) ((p "Oxente_2005_V1_2" 8) -5 -10 0 0 63 0) ((p "Oxente_2005_V1_2" 9) -2 0 0 0 0 0) ((p "Oxente_2005_V1_2" 10) -2 22 0 0 -84 0) ((p "Oxente_2005_V1_2" 11) -2 -22 0 0 0 0) ((p "UvA_Trilearn" 1 goalie) 50 -0 0 0 -179 0) ((p "UvA_Trilearn" 2) 16 -10 0 0 0 90))

(see_global 1 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 0 0 0 0) ((p "Oxente_2005_V1_2" 1 goalie) -45 0 0 0 0 0) ((p "Oxente_2005_V1_2" 2) -16 16 0 0 -45 0) ((p "Oxente_2005_V1_2" 3) -17 5 0 0 -16 0) ((p "Oxente_2005_V1_2" 4) -17 -5 0 0 17 0) ((p "Oxente_2005_V1_2" 5) -16 -16 0 0 46 0) ((p "Oxente_2005_V1_2" 6) -8 0 0 0 0 0) ((p "Oxente_2005_V1_2" 7) -5 10 0 0 -63 0) ((p "Oxente_2005_V1_2" 8) -5 -10 0 0 63 0) ((p "Oxente_2005_V1_2" 9) -2 0 0 0 0 0) ((p "Oxente_2005_V1_2" 10) -2 22 0 0 -84 0) ((p "Oxente_2005_V1_2" 11) -2 -22 0 0 85 0) ((p "UvA_Trilearn" 1 goalie) 50 0 0 0 -179 0) ((p "UvA_Trilearn" 2) 16 -10 0 0 146 1) ((p "UvA_Trilearn" 3) 21 0 0 0 -178 -1) ((p "UvA_Trilearn" 4) 15 0 0 0 -180 -1) ((p "UvA_Trilearn" 5) 16 10 0 0 -148 -1) ((p "UvA_Trilearn" 6) 2 11 0 0 -100 -1) ((p "UvA_Trilearn" 7) 2 -11 0 0 100 1) ((p "UvA_Trilearn" 8) 9.14534 -0.291964 0 0 179 0) ((p "UvA_Trilearn" 9) 9.22825 0.289614 -0 -0 -177 -1) ((p "UvA_Trilearn" 10) 2 -19 0 0 95 1) ((p "UvA_Trilearn" 11) 2 19 0 0 -95 -1))

(see_global 2 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 0 0 0 0) ((p "Oxente_2005_V1_2" 1 goalie) -45 0 0 0 0 0) ((p "Oxente_2005_V1_2" 2) -16 16 0 0 -45 0) ((p "Oxente_2005_V1_2" 3) -17 5 0 0 -16 0) ((p "Oxente_2005_V1_2" 4) -17 -5 0 0 17 0) ((p "Oxente_2005_V1_2" 5) -16 -16 0 0 46 0) ((p "Oxente_2005_V1_2" 6) -8 0 0 0 0 0) ((p "Oxente_2005_V1_2" 7) -5 10 0 0 -63 0) ((p "Oxente_2005_V1_2" 8) -5 -10 0 0 63 0) ((p "Oxente_2005_V1_2" 9) -1.36328 -0.03516 0.254688 -0.014064 0 0) ((p "Oxente_2005_V1_2" 10) -2 22 0 0 -84 0) ((p "Oxente_2005_V1_2" 11) -2 -22 0 0 85 0) ((p "UvA_Trilearn" 1 goalie) 50 0 0 0 -179 0) ((p "UvA_Trilearn" 2) 16 -10 0 0 146 1) ((p "UvA_Trilearn" 3) 21 0 0 0 -178 -1) ((p "UvA_Trilearn" 4) 15 0 0 0 -180 -1) ((p "UvA_Trilearn" 5) 16 10 0 0 -148 -1) ((p "UvA_Trilearn" 6) 2 11 0 0 -100 -1) ((p "UvA_Trilearn" 7) 2 -11 0 0 100 1) ((p "UvA_Trilearn" 8) 9.14514 -0.298224 0 0 179 0) ((p "UvA_Trilearn" 9) 9.22914 0.295852 -0 -0 -177 -1) ((p "UvA_Trilearn" 10) 2 -19 0 0 95 1) ((p "UvA_Trilearn" 11) 2 19 0 0 -95 -1))

(see_global 4 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 2.54115 0.25323 2.38868 0.238036) ((p "Oxente_2005_V1_2" 1 goalie) -45 0 0 0 0 0) ((p "Oxente_2005_V1_2" 2) -16 16 0 0 -45 0) ((p "Oxente_2005_V1_2" 3) -17 5 0 0 -16 0) ((p "Oxente_2005_V1_2" 4) -17 -5 0 0 17 0) ((p "Oxente_2005_V1_2" 5) -16 -16 0 0 46 0) ((p "Oxente_2005_V1_2" 6) -8 0 0 0 0 0) ((p "Oxente_2005_V1_2" 7) -5 10 0 0 -63 0) ((p "Oxente_2005_V1_2" 8) -5 -10 0 0 63 0) ((p "Oxente_2005_V1_2" 9) -0.237662 0.071211 0.12328 0.0152128 0 -21) ((p "Oxente_2005_V1_2" 10) -2 22 0 0 -4 -77) ((p "Oxente_2005_V1_2" 11) -2 -22 0 0 85 0) ((p "UvA_Trilearn" 1 goalie) 50 0 0 0 -179 0) ((p "UvA_Trilearn" 2) 16 -10 0 0 146 1) ((p "UvA_Trilearn" 3) 21 0 0 0 -178 -1) ((p "UvA_Trilearn" 4) 15 0 0 0 -

180 -1) ((p "UvA_Trilearn" 5) 16 10 0 0 -148 -1) ((p "UvA_Trilearn" 6) 2 11 0 0 -100 -1) ((p "UvA_Trilearn" 7) 2 -11 0 0 100 1) ((p "UvA_Trilearn" 8) 9.14513 -0.298261 -0 -0 178 0) ((p "UvA_Trilearn" 9) 9.22915 0.295889 -0 -0 -177 -1) ((p "UvA_Trilearn" 10) 2 -19 0 0 95 1) ((p "UvA_Trilearn" 11) 2 19 0 0 -95 -1))

(see_global 6 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 7.37135 1.01281 2.19551 0.394352) ((p "Oxente_2005_V1_2" 1 goalie) -43.5846 -0.0674058 0.343482 -0.0299863 0 0) ((p "Oxente_2005_V1_2" 2) -16 16 0 0 -65 32) ((p "Oxente_2005_V1_2" 3) -17 5 0 0 -103 90) ((p "Oxente_2005_V1_2" 4) -17 -5 0 0 43 -32) ((p "Oxente_2005_V1_2" 5) -16 -16 0 0 64 -28) ((p "Oxente_2005_V1_2" 6) -7.60659 -0.458463 0.157363 -0.183385 -45 51) ((p "Oxente_2005_V1_2" 7) -4.38468 10.044 0.246128 0.0175872 5 -43) ((p "Oxente_2005_V1_2" 8) -4.25889 -8.6492 0.180672 0.313365 63 -25) ((p "Oxente_2005_V1_2" 9) 0.533376 0.157393 0.256509 0.0281789 5 -1) ((p "Oxente_2005_V1_2" 10) -0.513534 21.8572 0.3637 -0.0235017 -4 -62) ((p "Oxente_2005_V1_2" 11) -1.35081 -21.8901 0.259678 0.043945 7 62) ((p "UvA_Trilearn" 1 goalie) 49.1687 0.00557478 -0.0999396 0.00280354 -102 -84) ((p "UvA_Trilearn" 2) 16 -10 0 0 5 90) ((p "UvA_Trilearn" 3) 21 0 0 0 21 89) ((p "UvA_Trilearn" 4) 15 0 0 0 17 -1) ((p "UvA_Trilearn" 5) 16 10 0 0 5 0) ((p "UvA_Trilearn" 6) 2 11 0 0 179 83) ((p "UvA_Trilearn" 7) 2 -11 0 0 -180 -90) ((p "UvA_Trilearn" 8) 8.13306 -0.328043 0.0257724 0.000491853 178 -90) ((p "UvA_Trilearn" 9) 8.08645 0.270204 0.0302775 0.00389523 -177 70) ((p "UvA_Trilearn" 10) 1.29004 -18.9681 -0.389769 0.0175108 -178 -90) ((p "UvA_Trilearn" 11) 1.39104 18.9642 -0.282559 -0.0166102 -178 72))

(see_global 7 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 7.58189 1.1446 0.197909 0.123885) ((p "Oxente_2005_V1_2" 1 goalie) -43.2244 -0.119114 0.144096 -0.0206832 7 -6) ((p "Oxente_2005_V1_2" 2) -16 16 0 0 -53 24) ((p "Oxente_2005_V1_2" 3) -17 5 0 0 -85 76) ((p "Oxente_2005_V1_2" 4) -16.5705 -4.62137 0.171792 0.151452 43 -32) ((p "Oxente_2005_V1_2" 5) -16 -16 0 0 52 -15) ((p "Oxente_2005_V1_2" 6) -7.06173 -1.14643 0.217946 -0.275188 -45 52) ((p "Oxente_2005_V1_2" 7) -3.55081 10.1261 0.333547 0.0328538 5 -40) ((p "Oxente_2005_V1_2" 8) -3.71522 -7.85585 0.217469 0.317343 63 -28) ((p "Oxente_2005_V1_2" 9) 1.42685 0.175747 0.35739 0.00734175 5 1) ((p "Oxente_2005_V1_2" 10) 0.496001 21.8447 0.403814 -0.00501217 -4 -62) ((p "Oxente_2005_V1_2" 11) -0.527841 -21.8586 0.329186 0.0126019 7 61) ((p "UvA_Trilearn" 1 goalie) 49.0619 0.00151976 -0.0427033 -0.00162201 134 37) ((p "UvA_Trilearn" 2) 16.5733 -9.96184 0.266018 0.0177069 5 90) ((p "UvA_Trilearn" 3) 21.5439 0.189918 0.21754 0.0759672 21 90) ((p "UvA_Trilearn" 4) 15 0 0 0 3 -1) ((p "UvA_Trilearn" 5) 16.5787 10.0173 0.268522 0.00802152 5 -90) ((p "UvA_Trilearn" 6) 2 11 0 0 12 -22) ((p "UvA_Trilearn" 7) 2 -11 0 0 -1 86) ((p "UvA_Trilearn" 8) 8.16024 -0.325541 0.0146037 0.00134482 -178 -1) ((p "UvA_Trilearn" 9) 8.11961 0.275332 0.0178198 0.00275607 -177 70) ((p "UvA_Trilearn" 10) 0.109881 -18.9453 -0.647907 0.0125351 -178 -83) ((p "UvA_Trilearn" 11) 0.499779 18.9974 -0.413543 0.0154067 -178 90))

(see_global 9 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 6.69898 -1.19838 -0.867858 -1.38745) ((p "Oxente_2005_V1_2" 1 goalie) -41.6929 -0.0993313 0.327316 0.00151626 7 -7) ((p "Oxente_2005_V1_2" 2) -15.1693 14.8767 0.198835 -0.25911 -53 23) ((p "Oxente_2005_V1_2" 3) -16.9009 3.55431 0.0274252 -0.325051 -85 78) ((p "Oxente_2005_V1_2" 4) -15.2951 -3.45208 0.25547 0.242261 43 -31) ((p

"Oxente_2005_V1_2" 5) -14.9831 -14.9329 0.240222 0.243282 52 -17) ((p "Oxente_2005_V1_2" 6) -6.29278 -1.83966 0.210852 -0.175275 -38 49) ((p "Oxente_2005_V1_2" 7) -1.5625 10.4027 0.415177 0.0517731 5 -53) ((p "Oxente_2005_V1_2" 8) -2.82522 -6.2657 0.164967 0.330197 63 -27) ((p "Oxente_2005_V1_2" 9) 3.20224 0.293686 0.364628 0.00575867 5 5) ((p "Oxente_2005_V1_2" 10) 2.36262 21.5861 0.370094 -0.0661158 -4 -72) ((p "Oxente_2005_V1_2" 11) 1.41699 -21.817 0.375941 0.00952036 7 67) ((p "UvA_Trilearn" 1 goalie) 49.0056 -0.00379709 -0.0066379 -0.000216428 88 90) ((p "UvA_Trilearn" 2) 17.7579 -9.93368 0.170426 0.00830026 -2 1) ((p "UvA_Trilearn" 3) 22.4217 0.289465 0.269026 0.0118836 0 1) ((p "UvA_Trilearn" 4) 16.3617 0.0402952 0.45303 0.0204591 3 -1) ((p "UvA_Trilearn" 5) 18.3924 9.98163 0.425341 -0.0119328 5 -1) ((p "UvA_Trilearn" 6) 3.10616 11.189 0.227234 0.024203 -4 -90) ((p "UvA_Trilearn" 7) 3.71743 -11.0639 0.567007 0.00456214 -1 37) ((p "UvA_Trilearn" 8) 8.18173 -0.324587 0.00395048 0.000260981 -179 66) ((p "UvA_Trilearn" 9) 8.14473 0.2789 0.00471281 0.000674519 -177 -3) ((p "UvA_Trilearn" 10) -2.27462 -18.9079 -0.603683 -0.0297034 -178 -90) ((p "UvA_Trilearn" 11) -1.55522 18.7279 -0.458967 -0.0795867 -178 78))

(see_global 12 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 4.11242 -5.04228 -0.755853 -1.07494) ((p "Oxente_2005_V1_2" 1 goalie) -41.1785 -0.120557 0.0207962 0.000477286 -74 65) ((p "Oxente_2005_V1_2" 2) -14.3505 13.744 0.0393972 -0.0430246 -69 24) ((p "Oxente_2005_V1_2" 3) -16.8251 2.22204 0.00375845 -0.057877 -106 89) ((p "Oxente_2005_V1_2" 4) -14.4594 -2.7852 0.187538 0.143796 37 -45) ((p "Oxente_2005_V1_2" 5) -14.0039 -13.9065 0.0777754 0.0903164 71 -42) ((p "Oxente_2005_V1_2" 6) -4.09586 -3.76386 0.292734 -0.276489 -38 29) ((p "Oxente_2005_V1_2" 7) 0.532888 10.4397 0.29872 -0.012259 -3 -73) ((p "Oxente_2005_V1_2" 8) -2.20812 -6.15236 0.167659 -0.148847 -50 60) ((p "Oxente_2005_V1_2" 9) 4.66069 0.521787 0.0619523 0.0144431 0 -84) ((p "Oxente_2005_V1_2" 10) 4.38726 21.3961 0.302825 -0.035535 -11 -79) ((p "Oxente_2005_V1_2" 11) 4.59946 -21.6027 0.430856 0.0469908 7 83) ((p "UvA_Trilearn" 1 goalie) 48.9961 -0.00493568 -0.00040191 -2.77308e-05 -89 -90) ((p "UvA_Trilearn" 2) 19.7446 -9.81452 0.194373 0.00662952 -9 1) ((p "UvA_Trilearn" 3) 23.7526 0.0903936 0.126994 -0.0229386 0 1) ((p "UvA_Trilearn" 4) 18.0985 0.0505669 0.230039 0.00932476 -13 -1) ((p "UvA_Trilearn" 5) 19.9177 9.92157 0.152829 -0.00593116 -61 -90) ((p "UvA_Trilearn" 6) 3.55331 11.2685 0.0374673 0.0039078 -179 57) ((p "UvA_Trilearn" 7) 3.71331 -11.0356 -0.000269979 0.00246925 126 72) ((p "UvA_Trilearn" 8) 8.18914 -0.323556 0.00061374 0.000110959 -146 -19) ((p "UvA_Trilearn" 9) 5.66364 0.0724553 -0.563364 -0.0405181 -177 36) ((p "UvA_Trilearn" 10) -5.65566 -19.1513 -0.610458 -0.032148 -178 -90) ((p "UvA_Trilearn" 11) -4.22432 18.5339 -0.240748 -0.0147619 175 90))

(see_global 15 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 1.81811 -8.04316 -0.695454 -0.887017) ((p "Oxente_2005_V1_2" 1 goalie) -40.9343 -1.41408 0.0630659 -0.29046 -80 71) ((p "Oxente_2005_V1_2" 2) -14.2796 13.6817 0.00341553 -0.00205567 -92 38) ((p "Oxente_2005_V1_2" 3) -17.3367 0.721872 -0.0994204 -0.258188 -114 90) ((p "Oxente_2005_V1_2" 4) -13.5567 -1.95473 0.0390208 0.0315152 -21 -2) ((p "Oxente_2005_V1_2" 5) -13.901 -13.7424 0.003186 0.00721669 20 0) ((p "Oxente_2005_V1_2" 6) -3.03719 -4.8574 0.076233 -0.102529 -58 25) ((p

"Oxente_2005_V1_2" 7) 0.970967 10.4422 0.0165416 7.26365e-05 -84 -1) ((p "Oxente_2005_V1_2" 8) -0.320786 -8.30022 0.276291 -0.291908 -50 55) ((p "Oxente_2005_V1_2" 9) 6.26296 0.336499 0.279505 -0.0528475 -10 -90) ((p "Oxente_2005_V1_2" 10) 7.21613 20.7929 0.375801 -0.0959144 -11 -90) ((p "Oxente_2005_V1_2" 11) 7.12136 -21.1187 0.183617 0.0474619 -3 90) ((p "UvA_Trilearn" 1 goalie) 49.0654 -0.671219 0.0046278 -0.0277298 -90 -81) ((p "UvA_Trilearn" 2) 20.0739 -9.82549 0.0181815 0.000751141 -136 -84) ((p "UvA_Trilearn" 3) 23.9631 0.0449507 0.00837543 -0.00142984 179 22) ((p "UvA_Trilearn" 4) 18.1518 -0.284373 -0.183 -0.205372 -133 -55) ((p "UvA_Trilearn" 5) 20.1919 9.92112 0.0161277 -0.00111923 179 46) ((p "UvA_Trilearn" 6) 3.6184 11.2728 0.00559684 0.000889479 -134 4) ((p "UvA_Trilearn" 7) 2.06444 -8.5153 -0.353571 0.542086 126 69) ((p "UvA_Trilearn" 8) 6.75282 -1.51594 -0.196911 -0.16724 -156 -7) ((p "UvA_Trilearn" 9) 2.17843 -0.0276058 -0.684012 0.0119637 -177 49) ((p "UvA_Trilearn" 10) -7.83628 -19.208 -0.307346 0.0123918 -160 -90) ((p "UvA_Trilearn" 11) -5.5258 18.4303 -0.150178 -0.0129242 -168 70))

(see_global 16 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 0.0583093 -9.84514 -1.65421 -1.69387) ((p "Oxente_2005_V1_2" 1 goalie) -40.8541 -1.6939 0.0320983 -0.111928 -12 1) ((p "Oxente_2005_V1_2" 2) -14.2763 13.6796 0.00134006 -0.000808872 -100 46) ((p "Oxente_2005_V1_2" 3) -17.4527 0.444704 -0.0463861 -0.110867 -84 59) ((p "Oxente_2005_V1_2" 4) -13.5211 -1.91965 0.014224 0.0140306 -24 -2) ((p "Oxente_2005_V1_2" 5) -13.8977 -13.7353 0.00130343 0.00285765 20 0) ((p "Oxente_2005_V1_2" 6) -2.68317 -5.41729 0.141609 -0.223956 -58 17) ((p "Oxente_2005_V1_2" 7) 0.988736 10.443 0.00710762 0.00031622 -86 -1) ((p "Oxente_2005_V1_2" 8) 0.392057 -8.99009 0.285137 -0.275949 -50 51) ((p "Oxente_2005_V1_2" 9) 7.17421 0.2545 0.3645 -0.0327993 -10 -90) ((p "Oxente_2005_V1_2" 10) 8.25661 20.5909 0.41619 -0.0807987 -11 -90) ((p "Oxente_2005_V1_2" 11) 7.82751 -21.1213 0.282461 -0.00103995 -3 90) ((p "UvA_Trilearn" 1 goalie) 49.071 -0.696571 0.00226495 -0.0101406 -90 -81) ((p "UvA_Trilearn" 2) 19.6886 -10.1758 -0.178803 -0.162555 -136 -13) ((p "UvA_Trilearn" 3) 23.9721 0.0439304 0.00360915 -0.000408122 179 22) ((p "UvA_Trilearn" 4) 17.55 -0.828 -0.323404 -0.292145 -133 14) ((p "UvA_Trilearn" 5) 20.2091 9.92028 0.00796482 -0.0003858 180 46) ((p "UvA_Trilearn" 6) 3.18906 10.7416 -0.235706 -0.291627 -134 73) ((p "UvA_Trilearn" 7) 1.72951 -7.98525 -0.183877 0.290997 126 -42) ((p "UvA_Trilearn" 8) 6.01178 -1.96458 -0.398234 -0.241104 -156 66) ((p "UvA_Trilearn" 9) 1.04953 -0.141994 -0.60667 -0.061472 -177 90) ((p "UvA_Trilearn" 10) -8.73453 -19.4896 -0.493142 -0.154604 -160 -90) ((p "UvA_Trilearn" 11) -6.22964 18.3528 -0.32658 -0.0359538 -168 70))

(see_global 19 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -4.79248 -14.8377 -1.39005 -1.40788) ((p "Oxente_2005_V1_2" 1 goalie) -40.8174 -1.88214 0.00131179 -0.00841991 -97 75) ((p "Oxente_2005_V1_2" 2) -14.2741 13.6785 8.46897e-05 -5.37309e-05 -123 49) ((p "Oxente_2005_V1_2" 3) -17.4265 -0.672426 -2.07775e-05 -0.0494118 -146 90) ((p "Oxente_2005_V1_2" 4) -13.5016 -1.89949 0.000663866 0.000822617 -171 89) ((p "Oxente_2005_V1_2" 5) -14.087 -14.2713 -0.0763006 -0.215992 -110 90) ((p "Oxente_2005_V1_2" 6) -1.48102 -7.22964 0.0985422 -0.115288 -97 -18) ((p "Oxente_2005_V1_2" 7) 0.538615 9.9961 -0.184187 -0.179047 -136 34) ((p "Oxente_2005_V1_2" 8) 1.14447 -8.68615 0.0571633 0.067347 -112 90) ((p

"Oxente_2005_V1_2" 9) 8.59662 -0.111351 0.122637 -0.0321491 -38 -90) ((p "Oxente_2005_V1_2" 10) 9.77036 20.2066 0.058271 -0.0166853 -39 -74) ((p "Oxente_2005_V1_2" 11) 9.68046 -21.3842 0.264025 -0.0569056 -12 90) ((p "UvA_Trilearn" 1 goalie) 49.1099 -1.0709 0.0146836 -0.144138 -90 -76) ((p "UvA_Trilearn" 2) 18.1774 -11.335 -0.338229 -0.256866 -148 -5) ((p "UvA_Trilearn" 3) 23.9781 0.042678 0.000262812 -7.99724e-05 -160 8) ((p "UvA_Trilearn" 4) 16.2461 -1.41189 -0.388341 -0.0787839 -170 57) ((p "UvA_Trilearn" 5) 19.5214 9.48391 -0.107155 -0.0685165 -155 22) ((p "UvA_Trilearn" 6) 1.67383 9.61248 -0.246233 -0.167472 -160 56) ((p "UvA_Trilearn" 7) 1.75991 -7.95494 0.00280025 0.00244598 -135 36) ((p "UvA_Trilearn" 8) 3.61266 -3.15238 -0.267107 -0.127169 175 89) ((p "UvA_Trilearn" 9) -1.92649 -0.490502 -0.335271 -0.0433392 -149 80) ((p "UvA_Trilearn" 10) -8.4181 -18.8899 0.180411 0.3095 57 -25) ((p "UvA_Trilearn" 11) -8.22836 17.9871 -0.317437 -0.106685 -160 61))

(see_global 21 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -7.32728 -17.7503 0.115672 0.13309) ((p "Oxente_2005_V1_2" 1 goalie) -40.9066 -3.35083 -0.014471 -0.360951 -97 73) ((p "Oxente_2005_V1_2" 2) -14.6507 12.9548 -0.0405379 -0.0766494 -134 55) ((p "Oxente_2005_V1_2" 3) -18.5189 -1.43461 -0.256765 -0.164486 -146 90) ((p "Oxente_2005_V1_2" 4) -14.8218 -2.19714 -0.315143 -0.0562932 -171 90) ((p "Oxente_2005_V1_2" 5) -14.4769 -15.1469 -0.126522 -0.263861 -120 90) ((p "Oxente_2005_V1_2" 6) -1.54958 -7.94126 -0.0609183 -0.244261 -104 -17) ((p "Oxente_2005_V1_2" 7) -0.185667 9.38741 -0.217682 -0.178843 -144 40) ((p "Oxente_2005_V1_2" 8) 1.29222 -8.20338 0.0829242 0.245725 -112 86) ((p "Oxente_2005_V1_2" 9) 8.77903 -0.165611 0.0206029 -0.00579149 -61 -73) ((p "Oxente_2005_V1_2" 10) 10.5394 19.6115 0.0883315 -0.0725479 -52 -62) ((p "Oxente_2005_V1_2" 11) 10.5297 -21.7205 0.239041 -0.109548 -20 -89) ((p "UvA_Trilearn" 1 goalie) 49.1363 -1.27587 0.00298876 -0.023927 -90 -75) ((p "UvA_Trilearn" 2) 16.9729 -12.2244 -0.185254 -0.127899 -141 -6) ((p "UvA_Trilearn" 3) 22.5124 -0.518372 -0.340529 -0.124871 -160 10) ((p "UvA_Trilearn" 4) 14.1606 -1.66938 -0.619509 -0.0583549 -170 27) ((p "UvA_Trilearn" 5) 18.0167 8.92692 -0.393108 -0.136467 -155 20) ((p "UvA_Trilearn" 6) -0.30391 8.77552 -0.585885 -0.270825 -160 56) ((p "UvA_Trilearn" 7) 0.854161 -8.8005 -0.296148 -0.273025 -135 2) ((p "UvA_Trilearn" 8) 2.82562 -2.94213 -0.2893 0.172776 132 69) ((p "UvA_Trilearn" 9) -3.53007 -1.25215 -0.442928 -0.216324 -149 12) ((p "UvA_Trilearn" 10) -7.67955 -17.595 -0.0134565 -0.0255566 88 68) ((p "UvA_Trilearn" 11) -10.1154 17.494 -0.474757 -0.126382 -160 61))

(see_global 22 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -7.21962 -17.6179 0.101206 0.124474) ((p "Oxente_2005_V1_2" 1 goalie) -40.9676 -4.36685 -0.0243994 -0.406407 -97 73) ((p "Oxente_2005_V1_2" 2) -15.1618 12.446 -0.204431 -0.203544 -134 58) ((p "Oxente_2005_V1_2" 3) -19.3154 -1.88025 -0.31859 -0.178256 -146 90) ((p "Oxente_2005_V1_2" 4) -15.7133 -2.26412 -0.356608 -0.026793 -171 90) ((p "Oxente_2005_V1_2" 5) -14.5803 -15.4383 -0.0413383 -0.116571 -41 29) ((p "Oxente_2005_V1_2" 6) -1.60103 -8.18421 -0.0205811 -0.0971808 -115 -7) ((p "Oxente_2005_V1_2" 7) -0.411519 9.22818 -0.0903408 -0.0636939 -153 48) ((p "Oxente_2005_V1_2" 8) 1.37436 -7.98177 0.0328585 0.0886424 -112 25) ((p "Oxente_2005_V1_2" 9) 8.80144 -0.169875 0.00896711 -0.00170537 -68 -65) ((p "Oxente_2005_V1_2" 10) 11.0248 19.0879 0.194147 -0.209419 -52 -64) ((p

"Oxente_2005_V1_2" 11) 10.7579 -21.8514 0.0913042 -0.0523597 -33 -90) ((p "UvA_Trilearn" 1 goalie) 49.138 -1.29917 0.000686238 -0.00932002 -90 -74) ((p "UvA_Trilearn" 2) 16.3397 -12.7426 -0.293822 -0.240486 -141 -5) ((p "UvA_Trilearn" 3) 21.6646 -0.871991 -0.339129 -0.141448 -160 11) ((p "UvA_Trilearn" 4) 12.987 -1.71068 -0.630706 -0.022194 -170 29) ((p "UvA_Trilearn" 5) 17.1113 8.50635 -0.420128 -0.195145 -155 21) ((p "UvA_Trilearn" 6) -1.50376 8.3815 -0.658719 -0.216316 -160 56) ((p "UvA_Trilearn" 7) 0.082279 -9.69415 -0.423763 -0.490616 -135 2) ((p "UvA_Trilearn" 8) 2.21774 -2.31401 -0.326677 0.337554 132 90) ((p "UvA_Trilearn" 9) -4.531 -1.79507 -0.537902 -0.291768 -149 84) ((p "UvA_Trilearn" 10) -7.69163 -17.6181 -0.00662967 -0.0127018 173 2) ((p "UvA_Trilearn" 11) -11.1866 17.0971 -0.497046 -0.184157 -160 62))

(see_global 25 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -9.34972 -16.5631 -0.638888 0.300461) ((p "Oxente_2005_V1_2" 1 goalie) -40.9321 -4.36675 0.00149754 -8.22192e-05 -28 0) ((p "Oxente_2005_V1_2" 2) -17.2202 10.3292 -0.286194 -0.310367 -134 61) ((p "Oxente_2005_V1_2" 3) -19.8738 -2.14495 -0.0236474 -0.0125025 -129 76) ((p "Oxente_2005_V1_2" 4) -17.0631 -2.6328 -0.117256 -0.0308476 -84 26) ((p "Oxente_2005_V1_2" 5) -14.1191 -15.7102 0.211479 -0.0429553 -14 2) ((p "Oxente_2005_V1_2" 6) -2.51723 -10.3173 -0.13674 -0.331611 -115 -12) ((p "Oxente_2005_V1_2" 7) -2.61574 7.95179 -0.310354 -0.206875 -153 49) ((p "Oxente_2005_V1_2" 8) 0.827737 -7.72041 -0.23248 0.0594018 162 67) ((p "Oxente_2005_V1_2" 9) 9.43683 -1.78571 -0.00501962 0.0125077 -79 -58) ((p "Oxente_2005_V1_2" 10) 11.6113 18.2473 0.122774 -0.212567 -61 -61) ((p "Oxente_2005_V1_2" 11) 11.6712 -22.4198 0.0441723 -0.0228141 -125 -61) ((p "UvA_Trilearn" 1 goalie) 49.1376 -1.31468 -3.46555e-05 -0.00064098 -90 -74) ((p "UvA_Trilearn" 2) 13.8079 -14.6672 -0.430926 -0.309992 -141 -1) ((p "UvA_Trilearn" 3) 19.0802 -1.87541 -0.34389 -0.156636 -160 -7) ((p "UvA_Trilearn" 4) 9.80387 -2.26043 0.0555484 0.0151614 -170 28) ((p "UvA_Trilearn" 5) 14.2642 7.07001 -0.440844 -0.255549 -155 22) ((p "UvA_Trilearn" 6) -5.15949 7.54518 -0.697463 -0.139601 -160 60) ((p "UvA_Trilearn" 7) -2.04163 -11.8517 -0.41049 -0.413532 -142 29) ((p "UvA_Trilearn" 8) 0.652409 -0.576075 -0.297859 0.322162 141 90) ((p "UvA_Trilearn" 9) -7.05293 -2.97165 -0.301355 -0.108895 -142 77) ((p "UvA_Trilearn" 10) -9.721 -17.4127 -0.678066 0.0529933 173 6) ((p "UvA_Trilearn" 11) -12.8446 16.6489 -0.355104 -0.0659527 -167 69))

(see_global 27 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -11.2838 -17.4085 -0.502235 -0.27641) ((p "Oxente_2005_V1_2" 1 goalie) -40.9302 -4.36681 0.000220588 -1.2395e-05 -22 0) ((p "Oxente_2005_V1_2" 2) -18.0447 9.60046 -0.21542 -0.160677 -146 73) ((p "Oxente_2005_V1_2" 3) -20.5344 -2.73932 -0.0812963 -0.0675418 -148 90) ((p "Oxente_2005_V1_2" 4) -17.7728 -2.72768 -0.236283 -0.0258653 -171 90) ((p "Oxente_2005_V1_2" 5) -12.5061 -16.0836 0.356354 -0.10535 -14 -17) ((p "Oxente_2005_V1_2" 6) -3.06924 -11.2069 -0.171587 -0.226322 -128 -10) ((p "Oxente_2005_V1_2" 7) -3.59904 7.41325 -0.259212 -0.122313 -161 56) ((p "Oxente_2005_V1_2" 8) -0.970073 -7.27265 -0.383359 0.0958353 162 64) ((p "Oxente_2005_V1_2" 9) 9.42953 -1.76927 -0.000916048 0.00174649 -141 -1) ((p "Oxente_2005_V1_2" 10) 12.4903 16.5781 0.169537 -0.357067 -61 -62) ((p "Oxente_2005_V1_2" 11) 12.2477 -22.6918 0.214151 -0.101109 -26 -90) ((p "UvA_Trilearn" 1 goalie) 49.1375 -1.3156 -1.52034e-05 -0.000110568 -90 -74) ((p

"UvA_Trilearn" 2) 12.79 -15.5333 -0.287863 -0.261521 -133 -5) ((p "UvA_Trilearn" 3) 17.7968 -2.44299 -0.144606 -0.066429 -167 -2) ((p "UvA_Trilearn" 4) 8.56486 -2.47688 -0.398811 -0.0463287 -170 26) ((p "UvA_Trilearn" 5) 12.9257 6.29254 -0.205116 -0.110797 -165 27) ((p "UvA_Trilearn" 6) -7.38041 7.03904 -0.643979 -0.159819 -160 60) ((p "UvA_Trilearn" 7) -3.36483 -12.6746 -0.469855 -0.256552 -152 -34) ((p "UvA_Trilearn" 8) -0.834961 0.776232 -0.429425 0.386549 141 63) ((p "UvA_Trilearn" 9) -8.72038 -4.0588 -0.459942 -0.352596 -142 4) ((p "UvA_Trilearn" 10) -10.6988 -17.4022 -0.183443 0.00146826 173 -90) ((p "UvA_Trilearn" 11) -13.7275 16.4244 -0.17237 -0.0418723 -167 69))

(see_global 31 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -10.8991 -9.80193 0.0713992 1.67838) ((p "Oxente_2005_V1_2" 1 goalie) -41.0704 -3.81102 -0.0562257 0.222316 98 -90) ((p "Oxente_2005_V1_2" 2) -19.747 8.62014 -0.304565 -0.167304 -157 89) ((p "Oxente_2005_V1_2" 3) -22.2593 -3.85516 -0.0407859 -0.0330382 -174 90) ((p "Oxente_2005_V1_2" 4) -18.9993 -2.95683 -0.0185836 -0.00307021 167 90) ((p "Oxente_2005_V1_2" 5) -10.9754 -16.6046 0.0272452 -0.0114355 165 -76) ((p "Oxente_2005_V1_2" 6) -4.76206 -12.9723 -0.102867 -0.0842218 -148 -42) ((p "Oxente_2005_V1_2" 7) -4.96236 7.04915 -0.0451082 -0.0156976 160 90) ((p "Oxente_2005_V1_2" 8) -3.8975 -5.73697 -0.227867 0.231554 127 82) ((p "Oxente_2005_V1_2" 9) 9.42779 -1.76656 -2.27127e-05 4.11319e-05 -156 0) ((p "Oxente_2005_V1_2" 10) 12.8132 15.9643 0.00615838 -0.00804139 -130 -1) ((p "Oxente_2005_V1_2" 11) 12.5648 -22.8701 0.0045336 -0.00284587 148 -1) ((p "UvA_Trilearn" 1 goalie) 49.1011 -0.566287 0.00178226 0.0912361 -90 -79) ((p "UvA_Trilearn" 2) 10.3275 -17.749 -0.218426 -0.268922 -118 -16) ((p "UvA_Trilearn" 3) 15.0964 -3.16986 -0.307094 -0.0477888 -177 -3) ((p "UvA_Trilearn" 4) 4.78767 -3.26437 -0.35435 -0.0751478 -178 22) ((p "UvA_Trilearn" 5) 10.6234 5.47938 -0.100907 -0.0399201 175 40) ((p "UvA_Trilearn" 6) -11.3741 5.8463 -0.287691 -0.101484 -168 54) ((p "UvA_Trilearn" 7) -5.57836 -13.9844 -0.121718 -0.0811788 -173 -44) ((p "UvA_Trilearn" 8) -3.50077 2.56416 -0.29067 0.188418 140 90) ((p "UvA_Trilearn" 9) -10.5589 -5.96222 -0.0400785 -0.3096 -77 -18) ((p "UvA_Trilearn" 10) -11.0361 -17.3962 -0.0139435 -0.0002172 -155 -2) ((p "UvA_Trilearn" 11) -15.5661 16.1743 -0.288857 0.0288306 176 89))

(see_global 32 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -10.8286 -8.08138 0.0663257 1.61731) ((p "Oxente_2005_V1_2" 1 goalie) -41.1129 -3.57334 -0.0170051 0.0950719 81 -87) ((p "Oxente_2005_V1_2" 2) -20.0742 8.42274 -0.130861 -0.0789587 -168 90) ((p "Oxente_2005_V1_2" 3) -22.3042 -3.89176 -0.0179562 -0.0146388 178 90) ((p "Oxente_2005_V1_2" 4) -19.6475 -2.8751 -0.259273 0.0326921 167 90) ((p "Oxente_2005_V1_2" 5) -11.5585 -16.4597 -0.23324 0.0579822 165 -82) ((p "Oxente_2005_V1_2" 6) -4.87352 -13.049 -0.0445823 -0.0306788 110 37) ((p "Oxente_2005_V1_2" 7) -5.01199 7.03798 -0.0198544 -0.00446793 153 90) ((p "Oxente_2005_V1_2" 8) -4.49946 -4.99275 -0.240785 0.297688 127 80) ((p "Oxente_2005_V1_2" 9) 9.42777 -1.76652 -8.59643e-06 1.53251e-05 -162 0) ((p "Oxente_2005_V1_2" 10) 12.8199 15.9554 0.00270482 -0.00353257 -133 -1) ((p "Oxente_2005_V1_2" 11) 12.5696 -22.8727 0.0019205 -0.00104286 149 -1) ((p "UvA_Trilearn" 1 goalie) 49.1095 -0.477004 0.00334831 0.0357133 -90 -79) ((p "UvA_Trilearn" 2) 10.125 -18.0023 -0.0939552 -0.117514 -110 -23) ((p "UvA_Trilearn" 3) 14.7724 -3.212 -0.1296 -0.016853 172 5) ((p

"UvA_Trilearn" 4) 4.43861 -3.31402 -0.187586 -0.0266801 175 22) ((p "UvA_Trilearn" 5) 10.5204 5.43674 -0.0478078 -0.0197817 167 44) ((p "UvA_Trilearn" 6) -12.3809 5.55974 -0.552732 -0.157319 -168 57) ((p "UvA_Trilearn" 7) -5.70447 -14.0695 -0.0692329 -0.0467359 177 -45) ((p "UvA_Trilearn" 8) -4.13798 3.09937 -0.342434 0.287627 140 90) ((p "UvA_Trilearn" 9) -10.5758 -6.2544 -0.00908988 -0.157017 -128 -67) ((p "UvA_Trilearn" 10) -11.6597 -17.7351 -0.342371 -0.186018 -155 -2) ((p "UvA_Trilearn" 11) -15.8644 16.2258 -0.138394 0.0239106 161 90))

(see_global 34 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -10.8179 -6.62047 0.000978648 0.0140807) ((p "Oxente_2005_V1_2" 1 goalie) -40.9462 -1.89129 0.0472997 0.3854 81 -87) ((p "Oxente_2005_V1_2" 2) -20.8888 8.28345 -0.279337 -0.0274106 -177 90) ((p "Oxente_2005_V1_2" 3) -23.236 -3.9406 -0.105406 -0.00969349 165 -89) ((p "Oxente_2005_V1_2" 4) -20.5604 -2.60725 -0.252266 0.0985368 150 -90) ((p "Oxente_2005_V1_2" 5) -12.6521 -16.2398 -0.133685 0.0265777 158 -79) ((p "Oxente_2005_V1_2" 6) -5.18755 -12.3449 -0.0299657 0.0807534 136 0) ((p "Oxente_2005_V1_2" 7) -5.55748 7.34517 -0.210366 0.124434 143 90) ((p "Oxente_2005_V1_2" 8) -5.39472 -4.21386 -0.272298 0.177686 152 49) ((p "Oxente_2005_V1_2" 9) 9.42775 -1.7665 -1.58578e-06 2.48515e-06 -177 5) ((p "Oxente_2005_V1_2" 10) 12.8242 15.9498 0.000508855 -0.000695788 -137 -1) ((p "Oxente_2005_V1_2" 11) 12.5725 -22.8743 0.000332406 -0.000157357 147 -1) ((p "UvA_Trilearn" 1 goalie) 49.1107 -0.428129 -0.00021235 0.00539375 -90 -83) ((p "UvA_Trilearn" 2) 9.9045 -18.7713 -0.0628445 -0.304038 -101 -35) ((p "UvA_Trilearn" 3) 13.8174 -3.11732 -0.110457 0.0118337 162 11) ((p "UvA_Trilearn" 4) 3.31024 -3.15373 -0.212519 0.036354 168 59) ((p "UvA_Trilearn" 5) 10.4431 5.41135 -0.0114052 -0.00317152 151 55) ((p "UvA_Trilearn" 6) -14.3419 4.9728 -0.380381 -0.111835 -179 90) ((p "UvA_Trilearn" 7) -5.92865 -14.1201 -0.0855352 -0.00576572 168 -10) ((p "UvA_Trilearn" 8) -5.30125 3.95585 -0.201161 0.160161 131 90) ((p "UvA_Trilearn" 9) -11.1082 -6.36752 0.0327405 0.000594275 177 -90) ((p "UvA_Trilearn" 10) -13.6066 -18.6216 -0.54147 -0.275011 -155 -90) ((p "UvA_Trilearn" 11) -17.0999 16.6546 -0.22292 0.0762475 161 90))

(see_global 36 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -11.9391 -6.74986 -1.05425 -0.135331) ((p "Oxente_2005_V1_2" 1 goalie) -40.8893 -1.37971 0.00530272 0.0566524 -10 -1) ((p "Oxente_2005_V1_2" 2) -22.2241 8.23539 -0.156954 0.00191296 -113 62) ((p "Oxente_2005_V1_2" 3) -24.7905 -3.56882 -0.333057 0.091641 165 -90) ((p "Oxente_2005_V1_2" 4) -20.915 -2.46836 -0.0430051 0.0132376 -23 -1) ((p "Oxente_2005_V1_2" 5) -14.0977 -15.6921 -0.31454 0.118901 158 -88) ((p "Oxente_2005_V1_2" 6) -5.85676 -12.2879 -0.25852 -0.012825 -175 -55) ((p "Oxente_2005_V1_2" 7) -6.34318 7.97513 -0.223916 0.195912 133 90) ((p "Oxente_2005_V1_2" 8) -6.30801 -4.24516 -0.250257 -0.0895531 -153 3) ((p "Oxente_2005_V1_2" 9) 7.88024 -1.92915 -0.369221 -0.0410528 -177 10) ((p "Oxente_2005_V1_2" 10) 12.8249 15.9488 7.46136e-05 -0.000116205 -136 -1) ((p "Oxente_2005_V1_2" 11) 12.5729 -22.8744 5.35556e-05 -1.81389e-05 159 -15) ((p "UvA_Trilearn" 1 goalie) 49.111 -0.421197 8.86268e-05 0.000747744 -90 -83) ((p "UvA_Trilearn" 2) 9.82162 -19.2323 -0.00998688 -0.0695493 -127 -6) ((p "UvA_Trilearn" 3) 12.3297 -2.71842 -0.32756 0.104529 162 15) ((p "UvA_Trilearn" 4) 2.73739 -3.04354 -0.141315 0.030685 168 -8) ((p "UvA_Trilearn" 5) 9.72519 5.88837

-0.102519 0.0637233 158 52) ((p "UvA_Trilearn" 6) -16.2352 4.9567 -0.419028 0.0144543 -179 82) ((p "UvA_Trilearn" 7) -7.25616 -13.9747 -0.252713 0.0115799 156 -75) ((p "UvA_Trilearn" 8) -6.36642 4.84502 -0.236362 0.177245 131 83) ((p "UvA_Trilearn" 9) -11.8939 -6.36036 -0.139916 -0.000963875 177 -19) ((p "UvA_Trilearn" 10) -15.1723 -19.0753 -0.554934 -0.121471 -164 -90) ((p "UvA_Trilearn" 11) -17.4554 16.7637 -0.0506377 0.0132205 167 90))

(see_global 39 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -14.8362 -7.22065 -0.878427 -0.140388) ((p "Oxente_2005_V1_2" 1 goalie) -40.8491 -1.84482 0.0137233 -0.214915 -92 82) ((p "Oxente_2005_V1_2" 2) -23.4164 8.34138 -0.11837 0.0107424 -117 57) ((p "Oxente_2005_V1_2" 3) -26.1982 -3.1748 -0.0530253 0.0160403 -25 13) ((p "Oxente_2005_V1_2" 4) -20.9856 -2.4472 -0.00285098 0.000770809 171 89) ((p "Oxente_2005_V1_2" 5) -16.9004 -14.6775 -0.362679 0.1149 158 -90) ((p "Oxente_2005_V1_2" 6) -7.93501 -12.5223 -0.150069 -0.0062583 -175 -53) ((p "Oxente_2005_V1_2" 7) -6.65995 8.32846 -0.0137442 0.0145594 161 86) ((p "Oxente_2005_V1_2" 8) -8.6967 -5.41535 -0.32205 -0.146127 -153 -5) ((p "Oxente_2005_V1_2" 9) 6.53313 -2.03933 -0.0525651 -0.00563866 -177 10) ((p "Oxente_2005_V1_2" 10) 12.825 15.9486 6.26001e-06 -6.80782e-06 170 54) ((p "Oxente_2005_V1_2" 11) 11.6893 -22.4422 -0.038169 0.0230817 157 -12) ((p "UvA_Trilearn" 1 goalie) 49.1111 -0.420111 8.76483e-08 4.86044e-05 -90 -83) ((p "UvA_Trilearn" 2) 9.20653 -20.1412 -0.0343833 -0.0581122 157 70) ((p "UvA_Trilearn" 3) 11.7982 -2.61828 -0.0213271 0.00151698 -180 9) ((p "UvA_Trilearn" 4) 2.02227 -2.84377 -0.130252 0.0258939 168 -8) ((p "UvA_Trilearn" 5) 8.18266 6.61219 -0.293865 0.101434 166 47) ((p "UvA_Trilearn" 6) -18.0569 4.95852 -0.529933 -0.031305 -179 74) ((p "UvA_Trilearn" 7) -8.52686 -13.6696 -0.414344 0.116182 164 -64) ((p "UvA_Trilearn" 8) -7.07892 5.453 -0.0733831 0.0599635 143 62) ((p "UvA_Trilearn" 9) -14.4351 -6.36552 -0.508946 -0.028556 177 -28) ((p "UvA_Trilearn" 10) -17.7954 -19.8596 -0.283217 -0.0850988 -173 -90) ((p "UvA_Trilearn" 11) -18.6756 17.1358 -0.0799613 0.0267258 175 90))

(see_global 41 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -16.5569 -7.08953 -0.804774 0.27671) ((p "Oxente_2005_V1_2" 1 goalie) -40.8104 -2.17642 0.00694187 -0.0390408 -14 0) ((p "Oxente_2005_V1_2" 2) -24.169 8.35382 -0.252411 -0.00200318 -175 90) ((p "Oxente_2005_V1_2" 3) -26.2741 -3.15484 -0.00921592 0.00219907 174 90) ((p "Oxente_2005_V1_2" 4) -21.5473 -2.24479 -0.223549 0.0807242 157 90) ((p "Oxente_2005_V1_2" 5) -17.6916 -13.932 -0.176381 0.256871 114 -35) ((p "Oxente_2005_V1_2" 6) -8.15972 -12.5257 -0.025784 -0.0017029 -177 -37) ((p "Oxente_2005_V1_2" 7) -6.68117 8.34906 -0.00222313 0.00237788 -118 2) ((p "Oxente_2005_V1_2" 8) -10.4345 -6.20611 -0.350116 -0.185991 -153 -11) ((p "Oxente_2005_V1_2" 9) 5.03734 -2.07036 -0.332715 -0.00594152 -177 10) ((p "Oxente_2005_V1_2" 10) 12.825 15.9486 1.20923e-06 -1.13889e-06 -92 1) ((p "Oxente_2005_V1_2" 11) 10.8529 -22.1458 -0.0995003 0.0386739 157 -6) ((p "UvA_Trilearn" 1 goalie) 49.1111 -0.420041 8.62497e-07 7.65149e-06 -90 -83) ((p "UvA_Trilearn" 2) 8.68366 -19.9449 -0.226984 0.11491 151 75) ((p "UvA_Trilearn" 3) 11.7681 -2.61428 -0.00368601 0.000485197 168 13) ((p "UvA_Trilearn" 4) 1.27413 -2.79808 -0.327284 0.00803956 177 52) ((p "UvA_Trilearn" 5) 7.04049 7.05007 -0.166651 0.0562958 172 40) ((p "UvA_Trilearn" 6) -19.1406 4.98182 -0.239228 0.000588499 -179 77) ((p "UvA_Trilearn" 7) -9.32868 -13.5007 -0.20597 0.0365707

173 1) ((p "UvA_Trilearn" 8) -7.85008 5.92985 -0.320581 0.189331 143 90) ((p "UvA_Trilearn" 9) -15.9586 -6.36157 -0.279619 -0.0061562 177 78) ((p "UvA_Trilearn" 10) -18.541 -20.0547 -0.181074 -0.0389357 -173 -90) ((p "UvA_Trilearn" 11) -20.3035 17.1641 -0.427039 0.0006936 175 90))

(see_global 43 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -19.0677 -9.84211 -1.15963 -1.26831) ((p "Oxente_2005_V1_2" 1 goalie) -40.797 -2.22691 0.00166415 -0.00538824 -1 1) ((p "Oxente_2005_V1_2" 2) -25.2566 8.35968 -0.122267 0.00485285 -106 34) ((p "Oxente_2005_V1_2" 3) -27.6571 -3.01954 -0.307148 0.045357 174 90) ((p "Oxente_2005_V1_2" 4) -21.8443 -2.15809 -0.0330842 0.00767418 165 90) ((p "Oxente_2005_V1_2" 5) -18.5119 -13.4175 -0.250236 0.0932993 166 -68) ((p "Oxente_2005_V1_2" 6) -9.62924 -12.6904 -0.331811 -0.0429758 -177 -42) ((p "Oxente_2005_V1_2" 7) -7.2203 8.53189 -0.214692 0.0721826 168 70) ((p "Oxente_2005_V1_2" 8) -10.9619 -6.47816 -0.0581469 -0.0353737 -170 1) ((p "Oxente_2005_V1_2" 9) 3.02679 -1.96626 -0.423293 0.0242748 -177 10) ((p "Oxente_2005_V1_2" 10) 12.825 15.9486 1.72451e-07 -1.36496e-07 175 0) ((p "Oxente_2005_V1_2" 11) 10.7264 -22.104 -0.013983 0.00412259 157 -6) ((p "UvA_Trilearn" 1 goalie) 49.1111 -0.420029 3.30545e-08 1.34361e-06 -90 -83) ((p "UvA_Trilearn" 2) 8.35555 -19.7839 -0.0514297 0.0266496 -157 32) ((p "UvA_Trilearn" 3) 10.9802 -2.36801 -0.0937363 0.0277332 -175 9) ((p "UvA_Trilearn" 4) -0.543422 -2.61661 -0.522716 0.0752832 177 59) ((p "UvA_Trilearn" 5) 5.86011 7.22652 -0.170913 0.0324292 -175 30) ((p "UvA_Trilearn" 6) -19.7016 4.98913 -0.110282 -0.00100273 122 90) ((p "UvA_Trilearn" 7) -10.7698 -13.2231 -0.328249 0.0553547 173 17) ((p "UvA_Trilearn" 8) -8.29039 6.18663 -0.0838697 0.0463299 159 90) ((p "UvA_Trilearn" 9) -16.3876 -6.33548 -0.07917 0.00888271 149 86) ((p "UvA_Trilearn" 10) -19.0184 -20.1873 -0.090431 -0.0292584 113 -27) ((p "UvA_Trilearn" 11) -21.0135 17.2016 -0.0992199 0.00303607 -169 72))

(see_global 44 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -20.194 -11.1297 -1.05872 -1.2103) ((p "Oxente_2005_V1_2" 1 goalie) -40.7952 -2.23238 0.000710775 -0.00218597 -81 76) ((p "Oxente_2005_V1_2" 2) -25.3732 8.36828 -0.046626 0.00343887 -74 2) ((p "Oxente_2005_V1_2" 3) -28.5316 -2.87143 -0.349767 0.0592432 174 90) ((p "Oxente_2005_V1_2" 4) -21.8761 -2.15058 -0.0127446 0.00300446 176 90) ((p "Oxente_2005_V1_2" 5) -19.2789 -13.2384 -0.306778 0.0716682 166 -56) ((p "Oxente_2005_V1_2" 6) -9.97651 -12.7207 -0.138907 -0.0121314 173 -2) ((p "Oxente_2005_V1_2" 7) -7.4243 8.58428 -0.0816005 0.0209545 177 57) ((p "Oxente_2005_V1_2" 8) -11.0162 -6.50924 -0.0217233 -0.0124343 135 70) ((p "Oxente_2005_V1_2" 9) 2.60604 -1.92587 -0.1683 0.0161545 -158 1) ((p "Oxente_2005_V1_2" 10) 12.825 15.9486 6.70275e-08 -6.26918e-08 -141 0) ((p "Oxente_2005_V1_2" 11) 10.7132 -22.1008 -0.00529582 0.00127 160 -1) ((p "UvA_Trilearn" 1 goalie) 49.1111 -0.420028 1.52647e-08 5.70452e-07 -90 -83) ((p "UvA_Trilearn" 2) 8.30662 -19.7551 -0.0227023 0.0133706 -150 28) ((p "UvA_Trilearn" 3) 10.877 -2.3357 -0.0412639 0.0129232 -177 13) ((p "UvA_Trilearn" 4) -1.02442 -2.56277 -0.258487 0.0289347 -174 54) ((p "UvA_Trilearn" 5) 5.67754 7.26772 -0.0847116 0.0191154 -176 31) ((p "UvA_Trilearn" 6) -19.8024 4.97844 -0.0553016 -0.00586657 33 -89) ((p "UvA_Trilearn" 7) -11.1202 -13.1365 -0.192343 0.047532 -176 14) ((p "UvA_Trilearn" 8) -8.91022 6.37044 -0.333095 0.0987824 159 90) ((p "UvA_Trilearn" 9) -16.4722 -6.32373 -0.0454572 0.00631483 -170 41) ((p

"UvA_Trilearn" 10) -19.1179 -20.2096 -0.0546038 -0.0122015 93 -74) ((p "UvA_Trilearn" 11) -21.1097 17.212 -0.0446102 0.00480794 -161 63))

(see_global 49 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -24.8036 -16.5757 -0.760795 -0.907124) ((p "Oxente_2005_V1_2" 1 goalie) -41.1262 -5.00346 -0.0180442 -0.155025 -83 47) ((p "Oxente_2005_V1_2" 2) -27.6987 7.60218 -0.380737 -0.139022 -164 81) ((p "Oxente_2005_V1_2" 3) -31.2934 -3.74104 -0.171779 -0.185675 -126 64) ((p "Oxente_2005_V1_2" 4) -25.3005 -2.07772 -0.346692 0.00209911 -176 90) ((p "Oxente_2005_V1_2" 5) -22.3093 -14.8917 -0.284614 -0.194881 -141 -6) ((p "Oxente_2005_V1_2" 6) -11.0571 -12.9333 -0.101146 -0.0199656 -141 -24) ((p "Oxente_2005_V1_2" 7) -10.9913 8.40986 -0.40299 -0.0158338 -175 56) ((p "Oxente_2005_V1_2" 8) -12.3008 -5.97261 -0.282025 0.144851 157 63) ((p "Oxente_2005_V1_2" 9) 2.32998 -1.88373 -0.00200126 0.000153522 -152 1) ((p "Oxente_2005_V1_2" 10) 10.8208 15.9162 -0.0549331 -0.00410742 -138 1) ((p "Oxente_2005_V1_2" 11) 10.1471 -22.2455 -0.223234 -0.0587663 -163 -27) ((p "UvA_Trilearn" 1 goalie) 49.1569 -0.937861 0.000825529 -0.00817281 -90 -80) ((p "UvA_Trilearn" 2) 7.06937 -20.8492 -0.176092 -0.250305 -119 15) ((p "UvA_Trilearn" 3) 9.47797 -2.94119 -0.264105 -0.0523006 -173 1) ((p "UvA_Trilearn" 4) -2.11383 -2.67299 -0.329384 -0.074935 -167 19) ((p "UvA_Trilearn" 5) 2.97516 6.49509 -0.325226 -0.137355 -159 19) ((p "UvA_Trilearn" 6) -21.9964 4.66876 -0.517576 -0.0889591 -163 90) ((p "UvA_Trilearn" 7) -13.552 -13.7317 -0.422429 -0.161755 -157 26) ((p "UvA_Trilearn" 8) -11.7566 6.80965 -0.306451 0.0244591 -174 57) ((p "UvA_Trilearn" 9) -20.367 -7.01491 -0.471288 -0.0877282 -163 82) ((p "UvA_Trilearn" 10) -19.973 -18.921 -0.384796 -0.125892 -157 -47) ((p "UvA_Trilearn" 11) -23.9063 16.8352 -0.373416 -0.108879 -164 74))

(see_global 52 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -26.951 -19.1647 -0.621622 -0.783853) ((p "Oxente_2005_V1_2" 1 goalie) -40.7706 -7.01777 0.0228441 -0.12952 -100 58) ((p "Oxente_2005_V1_2" 2) -30.452 6.65536 -0.372087 -0.146314 -164 81) ((p "Oxente_2005_V1_2" 3) -32.1532 -4.31434 -0.23462 -0.122005 -156 85) ((p "Oxente_2005_V1_2" 4) -27.3234 -2.38965 -0.367721 -0.0511205 -170 80) ((p "Oxente_2005_V1_2" 5) -23.8097 -16.0884 -0.214771 -0.21056 -132 -4) ((p "Oxente_2005_V1_2" 6) -12.6091 -13.3637 -0.321827 -0.0844949 -161 2) ((p "Oxente_2005_V1_2" 7) -12.9101 8.05935 -0.334573 -0.0981081 -166 47) ((p "Oxente_2005_V1_2" 8) -14.1512 -5.2257 -0.293835 0.0521528 178 51) ((p "Oxente_2005_V1_2" 9) 2.32708 -1.8838 -0.000124909 3.25698e-06 -129 -19) ((p "Oxente_2005_V1_2" 10) 10.1931 15.6518 -0.222894 -0.100236 -156 20) ((p "Oxente_2005_V1_2" 11) 9.02443 -22.5964 -0.0427915 -0.0139689 135 -1) ((p "UvA_Trilearn" 1 goalie) 49.1575 -0.950879 -4.67731e-05 -0.000473511 -90 -78) ((p "UvA_Trilearn" 2) 6.36637 -21.6529 -0.219592 -0.197561 -147 43) ((p "UvA_Trilearn" 3) 6.71234 -3.21497 -0.402357 -0.0215806 -173 -3) ((p "UvA_Trilearn" 4) -5.16278 -3.41672 -0.629767 -0.128258 -167 22) ((p "UvA_Trilearn" 5) 0.804899 5.62398 -0.218289 -0.102861 -159 20) ((p "UvA_Trilearn" 6) -23.2566 4.33777 -0.141333 -0.0430083 -155 58) ((p "UvA_Trilearn" 7) -14.5576 -14.1919 -0.0965105 -0.0389583 -150 26) ((p "UvA_Trilearn" 8) -13.2786 6.77922 -0.406718 -0.000692447 -174 57) ((p "UvA_Trilearn" 9) -22.1986 -7.39152 -0.156433 -0.0395336 -148 70) ((p "UvA_Trilearn" 10) -23.2302 -20.402 -0.551924 -0.281676 -157 -42) ((p "UvA_Trilearn" 11) -25.3065 16.1684 -0.298161 -0.191514 -147 55))

(see_global 54 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -28.1303 -20.7026 -0.554425 -0.678468) ((p "Oxente_2005_V1_2" 1 goalie) -40.988 -8.53877 -0.0586657 -0.340737 -100 57) ((p "Oxente_2005_V1_2" 2) -32.3912 6.05645 -0.363156 -0.109391 -164 82) ((p "Oxente_2005_V1_2" 3) -33.8181 -5.09862 -0.363712 -0.155064 -156 85) ((p "Oxente_2005_V1_2" 4) -29.1522 -2.70924 -0.360334 -0.0817712 -170 83) ((p "Oxente_2005_V1_2" 5) -24.951 -17.396 -0.233465 -0.253283 -132 -3) ((p "Oxente_2005_V1_2" 6) -14.3921 -13.8588 -0.385188 -0.0977443 -161 6) ((p "Oxente_2005_V1_2" 7) -14.8343 7.63263 -0.376357 -0.085918 -166 50) ((p "Oxente_2005_V1_2" 8) -15.374 -5.09109 -0.13232 0.00742764 -175 46) ((p "Oxente_2005_V1_2" 9) 1.83216 -2.50898 -0.0607415 -0.0654828 -129 -19) ((p "Oxente_2005_V1_2" 10) 8.66078 14.9665 -0.334881 -0.150498 -156 19) ((p "Oxente_2005_V1_2" 11) 8.42075 -22.9918 -0.222666 -0.152307 -147 -29) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951575 -3.6614e-06 -8.50205e-05 -90 -77) ((p "UvA_Trilearn" 2) 5.5437 -22.2034 -0.278431 -0.162876 -157 62) ((p "UvA_Trilearn" 3) 5.36113 -3.26922 -0.154313 -0.000873834 -162 -11) ((p "UvA_Trilearn" 4) -7.06149 -3.74258 -0.332598 -0.0618948 -159 17) ((p "UvA_Trilearn" 5) 0.28523 5.45306 -0.0793508 -0.0276 -153 15) ((p "UvA_Trilearn" 6) -24.2846 3.94145 -0.443527 -0.159248 -155 57) ((p "UvA_Trilearn" 7) -15.8346 -14.8432 -0.264196 -0.141256 -150 -39) ((p "UvA_Trilearn" 8) -14.4652 6.65663 -0.400589 -0.0800628 -165 48) ((p "UvA_Trilearn" 9) -22.6948 -7.61156 -0.130844 -0.0604664 -148 1) ((p "UvA_Trilearn" 10) -25.3961 -21.3145 -0.552414 -0.231938 -157 -38) ((p "UvA_Trilearn" 11) -26.0267 15.8076 -0.130644 -0.0639053 -147 54))

(see_global 60 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -31.2459 -24.2713 -0.413859 -0.483463) ((p "Oxente_2005_V1_2" 1 goalie) -42.2709 -9.95681 -0.282441 -0.0665041 -177 90) ((p "Oxente_2005_V1_2" 2) -36.9803 4.85817 -0.0702223 -0.01944 -71 0) ((p "Oxente_2005_V1_2" 3) -38.1866 -7.70269 -0.298559 -0.245613 -141 76) ((p "Oxente_2005_V1_2" 4) -33.6884 -3.96288 -0.352435 -0.160881 -162 81) ((p "Oxente_2005_V1_2" 5) -28.7808 -21.8645 -0.214172 -0.306693 -132 -3) ((p "Oxente_2005_V1_2" 6) -20.0966 -16.1228 -0.401649 -0.125152 -161 16) ((p "Oxente_2005_V1_2" 7) -18.7688 5.85498 -0.118956 -0.0651768 -164 51) ((p "Oxente_2005_V1_2" 8) -20.4268 -5.47912 -0.16671 0.0132191 -175 54) ((p "Oxente_2005_V1_2" 9) 1.12658 -3.44208 -0.00184086 -0.00186718 -146 -1) ((p "Oxente_2005_V1_2" 10) 6.86204 13.9449 -0.306354 -0.201916 -148 13) ((p "Oxente_2005_V1_2" 11) 7.5248 -23.4985 -0.212526 -0.104421 -150 -31) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 3.19405e-08 -3.52553e-07 -90 -75) ((p "UvA_Trilearn" 2) 4.29384 -23.1229 -0.0441116 -0.0388889 -113 28) ((p "UvA_Trilearn" 3) 3.30792 -3.69114 -0.00850424 -0.00146631 178 14) ((p "UvA_Trilearn" 4) -8.73301 -4.19134 -0.108465 -0.0413782 -159 20) ((p "UvA_Trilearn" 5) -2.2775 3.8415 -0.125135 -0.0779017 -153 17) ((p "UvA_Trilearn" 6) -28.8319 2.60288 -0.244729 -0.0612524 -167 72) ((p "UvA_Trilearn" 7) -18.3102 -16.229 -0.22008 -0.0898417 -150 -33) ((p "UvA_Trilearn" 8) -16.9521 5.82971 -0.174573 -0.0860322 -156 42) ((p "UvA_Trilearn" 9) -25.5406 -8.72194 -0.468309 -0.130706 -165 21) ((p "UvA_Trilearn" 10) -31.5685 -23.106 -0.679308 0.0445886 174 -75) ((p "UvA_Trilearn" 11) -28.6944 14.446 -0.135183 -0.0509141 -155 61))

(see_global 62 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -33.874 -23.5789 -1.1786 0.285555) ((p "Oxente_2005_V1_2" 1 goalie) -43.224 -10.0127 -0.276098 -0.00305351 177 90)

((p "Oxente_2005_V1_2" 2) -36.5291 4.5417 0.206308 -0.119162 -25 -57) ((p "Oxente_2005_V1_2" 3) -39.0426 -8.29482 -0.236766 -0.142194 -149 83) ((p "Oxente_2005_V1_2" 4) -34.8319 -4.19167 -0.317221 -0.0411439 -169 82) ((p "Oxente_2005_V1_2" 5) -30.237 -23.5808 -0.301561 -0.351928 -132 -4) ((p "Oxente_2005_V1_2" 6) -21.4488 -16.5343 -0.145801 -0.0477188 -146 1) ((p "Oxente_2005_V1_2" 7) -20.3859 5.35751 -0.353038 -0.101893 -164 51) ((p "Oxente_2005_V1_2" 8) -20.6372 -5.43701 -0.0239985 0.00609744 -122 1) ((p "Oxente_2005_V1_2" 9) 1.12388 -3.44478 -0.000304678 -0.000311244 -148 -1) ((p "Oxente_2005_V1_2" 10) 5.8623 13.2584 -0.274827 -0.189998 -148 13) ((p "Oxente_2005_V1_2" 11) 6.04746 -24.4251 -0.315071 -0.207817 -150 -30) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 9.46998e-11 -5.58913e-08 -90 -75) ((p "UvA_Trilearn" 2) 4.1254 -23.2171 0.0310939 0.143298 -113 26) ((p "UvA_Trilearn" 3) 3.29632 -3.69369 -0.00135481 -0.000331309 179 13) ((p "UvA_Trilearn" 4) -9.19773 -4.34391 -0.128405 -0.0453753 -159 18) ((p "UvA_Trilearn" 5) -3.22227 3.31633 -0.137011 -0.0751518 165 58) ((p "UvA_Trilearn" 6) -29.5153 2.43681 -0.171064 -0.0409603 -167 68) ((p "UvA_Trilearn" 7) -18.6352 -16.3398 -0.0618374 -0.021862 -155 35) ((p "UvA_Trilearn" 8) -17.498 5.57849 -0.143804 -0.0646255 -156 39) ((p "UvA_Trilearn" 9) -27.5864 -9.24845 -0.558393 -0.123998 -165 88) ((p "UvA_Trilearn" 10) -33.4088 -22.9657 -0.611418 0.0357375 174 -62) ((p "UvA_Trilearn" 11) -29.5683 14.0167 -0.30031 -0.159263 -155 59))

(see_global 64 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -36.105 -22.9572 0.0982844 -0.0272835) ((p "Oxente_2005_V1_2" 1 goalie) -44.4478 -10.0368 -0.143208 0.00365984 173 90) ((p "Oxente_2005_V1_2" 2) -36.2485 4.34168 0.0314588 -0.0256133 -137 57) ((p "Oxente_2005_V1_2" 3) -39.4075 -8.47704 -0.0444521 -0.017582 -173 90) ((p "Oxente_2005_V1_2" 4) -35.9162 -4.31132 -0.303205 -0.0300976 -176 84) ((p "Oxente_2005_V1_2" 5) -31.1508 -23.9492 -0.262395 -0.0142109 169 0) ((p "Oxente_2005_V1_2" 6) -21.6522 -16.5976 -0.0226925 -0.0067691 -176 22) ((p "Oxente_2005_V1_2" 7) -20.8939 5.19533 -0.0545404 -0.0186775 -172 59) ((p "Oxente_2005_V1_2" 8) -21.1904 -5.56966 -0.211118 -0.0549501 -162 39) ((p "Oxente_2005_V1_2" 9) 1.12339 -3.44521 -6.01304e-05 -4.6597e-05 -150 0) ((p "Oxente_2005_V1_2" 10) 5.45318 12.9944 -0.0494946 -0.0279361 -137 0) ((p "Oxente_2005_V1_2" 11) 5.56531 -24.7097 -0.0573493 -0.0290568 -167 -14) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -2.9763e-10 -9.46352e-09 -90 -75) ((p "UvA_Trilearn" 2) 3.62323 -23.2507 -0.242124 -0.0875205 -148 43) ((p "UvA_Trilearn" 3) 3.29454 -3.69417 -0.000194052 -5.43352e-05 179 13) ((p "UvA_Trilearn" 4) -10.0895 -4.74013 -0.348958 -0.158323 -159 18) ((p "UvA_Trilearn" 5) -3.40543 3.20276 -0.0254026 -0.018448 177 44) ((p "UvA_Trilearn" 6) -29.7783 2.38924 -0.0493197 -0.00882476 23 -89) ((p "UvA_Trilearn" 7) -19.7827 -16.964 -0.232159 -0.131156 -169 42) ((p "UvA_Trilearn" 8) -17.8482 5.38543 -0.0638901 -0.0329655 -164 45) ((p "UvA_Trilearn" 9) -29.1836 -9.59188 -0.308354 -0.0499056 -174 90) ((p "UvA_Trilearn" 10) -35.7259 -22.8901 0.0618748 -0.003223 174 -68) ((p "UvA_Trilearn" 11) -30.2563 13.5957 -0.124943 -0.0823609 -155 59))

(see_global 65 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -35.9915 -22.9938 -0.00912751 0.00282065) ((p "Oxente_2005_V1_2" 1 goalie) -44.5986 -10.042 -0.0603201 -0.00210023 -176 90) ((p "Oxente_2005_V1_2" 2) -36.216 4.31527 0.0129989 -

0.0105634 178 90) ((p "Oxente_2005_V1_2" 3) -40.0423 -8.55087 -0.253921 -0.0295313 -173 90) ((p "Oxente_2005_V1_2" 4) -36.7585 -4.3039 -0.336908 0.00296897 -176 87) ((p "Oxente_2005_V1_2" 5) -31.9347 -23.921 -0.313528 0.0112812 169 -1) ((p "Oxente_2005_V1_2" 6) -22.2419 -16.6105 -0.235889 -0.0051846 -176 20) ((p "Oxente_2005_V1_2" 7) -21.5509 5.15484 -0.262797 -0.0161964 -172 61) ((p "Oxente_2005_V1_2" 8) -21.4086 -5.63172 -0.0872744 -0.0248247 -170 42) ((p "Oxente_2005_V1_2" 9) 1.12333 -3.44526 -2.45268e-05 -1.99107e-05 -153 0) ((p "Oxente_2005_V1_2" 10) 5.40257 12.9707 -0.0202434 -0.00949216 -138 0) ((p "Oxente_2005_V1_2" 11) 5.50275 -24.7379 -0.0250227 -0.011309 -167 -14) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -7.66344e-11 -3.80889e-09 -90 -75) ((p "UvA_Trilearn" 2) 2.89222 -23.659 -0.339193 -0.189483 -148 47) ((p "UvA_Trilearn" 3) 3.29435 -3.69421 -7.68955e-05 -1.56564e-05 179 11) ((p "UvA_Trilearn" 4) -10.4191 -4.90122 -0.177151 -0.0865656 -168 22) ((p "UvA_Trilearn" 5) -3.42969 3.18417 -0.0112536 -0.00862689 179 39) ((p "UvA_Trilearn" 6) -29.1253 2.70034 0.358516 0.170794 23 -90) ((p "UvA_Trilearn" 7) -19.9993 -17.0827 -0.118906 -0.065183 -175 46) ((p "UvA_Trilearn" 8) -18.4338 5.24053 -0.314685 -0.0778698 -164 41) ((p "UvA_Trilearn" 9) -30.0837 -9.78524 -0.483701 -0.10391 -174 90) ((p "UvA_Trilearn" 10) -36.3485 -22.8496 0.0332755 -0.00185906 174 -69) ((p "UvA_Trilearn" 11) -30.3799 13.5074 -0.0573207 -0.0409651 171 90))

(see_global 67 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -37.3383 -23.104 0.126772 0.00633272) ((p "Oxente_2005_V1_2" 1 goalie) -45.3814 -10.0581 -0.0847891 -0.00614853 -92 38) ((p "Oxente_2005_V1_2" 2) -36.9959 4.27224 -0.094003 -0.0098525 -93 9) ((p "Oxente_2005_V1_2" 3) -41.7849 -8.82045 -0.350614 -0.0754414 -173 90) ((p "Oxente_2005_V1_2" 4) -38.5058 -4.41108 -0.362268 -0.0220376 -176 90) ((p "Oxente_2005_V1_2" 5) -33.7148 -23.5432 -0.355355 0.0777741 169 -3) ((p "Oxente_2005_V1_2" 6) -23.3138 -16.6158 -0.117596 -0.00799562 -176 21) ((p "Oxente_2005_V1_2" 7) -23.4624 5.07865 -0.403701 -0.0133273 -172 57) ((p "Oxente_2005_V1_2" 8) -22.9961 -5.97568 -0.337693 -0.0619284 -170 42) ((p "Oxente_2005_V1_2" 9) 1.1233 -3.44529 -3.32061e-06 -3.01976e-06 -153 0) ((p "Oxente_2005_V1_2" 10) 5.37148 12.9568 -0.0036652 -0.0017961 -138 0) ((p "Oxente_2005_V1_2" 11) 5.46855 -24.7517 -0.00395074 -0.00160177 179 -1) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -3.00681e-11 -6.30168e-10 -90 -76) ((p "UvA_Trilearn" 2) 1.72238 -24.3921 -0.178944 -0.100007 171 10) ((p "UvA_Trilearn" 3) 3.29425 -3.69422 -1.23129e-05 -1.81045e-06 179 11) ((p "UvA_Trilearn" 4) -10.9833 -5.05646 -0.139412 -0.0403372 -168 24) ((p "UvA_Trilearn" 5) -3.44627 3.17028 -0.00257557 -0.00215044 180 39) ((p "UvA_Trilearn" 6) -29.1322 2.72345 -0.00123467 0.00462357 177 67) ((p "UvA_Trilearn" 7) -21.3263 -17.2266 -0.263219 -0.0296551 -168 44) ((p "UvA_Trilearn" 8) -19.8895 4.87699 -0.293408 -0.0713099 -164 43) ((p "UvA_Trilearn" 9) -31.7973 -10.0126 -0.354955 -0.0488226 -174 90) ((p "UvA_Trilearn" 10) -37.4288 -22.7297 0.0217922 -0.000182335 174 -72) ((p "UvA_Trilearn" 11) -31.4917 13.6092 -0.199476 0.0291358 171 90))

(see_global 69 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -37.9847 -22.7949 -0.724533 0.287994) ((p "Oxente_2005_V1_2" 1 goalie) -45.49 -10.065 -0.0122891 3.30416e-05 -58 1) ((p "Oxente_2005_V1_2" 2) -37.738 4.19814 -0.261895 -0.0261085 -176

89) ((p "Oxente_2005_V1_2" 3) -43.7382 -9.09175 -0.38532 -0.0460813 -173 90) ((p "Oxente_2005_V1_2" 4) -39.7787 -4.63269 -0.144583 -0.032283 170 90) ((p "Oxente_2005_V1_2" 5) -35.4926 -23.0286 -0.35007 0.102355 169 12) ((p "Oxente_2005_V1_2" 6) -23.4646 -16.621 -0.0173174 0.000125598 -176 21) ((p "Oxente_2005_V1_2" 7) -24.0063 5.08385 -0.0616205 -0.00260651 -115 0) ((p "Oxente_2005_V1_2" 8) -25.0014 -6.22191 -0.408158 -0.0298089 -170 43) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -5.91781e-07 -4.17998e-07 -153 0) ((p "Oxente_2005_V1_2" 10) 5.36609 12.9536 -0.000572262 -0.000374696 -176 34) ((p "Oxente_2005_V1_2" 11) 4.81806 -24.875 -0.25868 -0.048768 -170 -12) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -7.16397e-12 -9.98661e-11 -90 -76) ((p "UvA_Trilearn" 2) 1.48277 -24.5501 -0.034657 -0.0212841 -12 -73) ((p "UvA_Trilearn" 3) 3.29423 -3.69422 -2.12476e-06 -4.49963e-07 179 11) ((p "UvA_Trilearn" 4) -12.0639 -5.24135 -0.224929 -0.0490575 -168 23) ((p "UvA_Trilearn" 5) -3.45 3.16756 -0.000560705 -0.00037498 180 38) ((p "UvA_Trilearn" 6) -29.1332 2.73078 -0.000119111 0.00142817 -172 63) ((p "UvA_Trilearn" 7) -22.9236 -17.4434 -0.359735 -0.0392288 -168 -25) ((p "UvA_Trilearn" 8) -20.5039 4.80907 -0.156368 -0.0137275 -173 52) ((p "UvA_Trilearn" 9) -32.5319 -10.0734 -0.155662 -0.0189575 -174 30) ((p "UvA_Trilearn" 10) -37.3969 -22.7337 0.00626672 -0.000880875 143 -73) ((p "UvA_Trilearn" 11) -32.7809 13.7128 -0.235374 0.0110479 171 90))

(see_global 72 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -36.9864 -18.6755 0.759504 1.74685) ((p "Oxente_2005_V1_2" 1 goalie) -45.5095 -10.0648 -0.000829926 7.27203e-05 -56 1) ((p "Oxente_2005_V1_2" 2) -39.8931 4.11297 -0.160176 -0.000564095 176 90) ((p "Oxente_2005_V1_2" 3) -44.3079 -9.1537 -0.0235353 -0.0051544 15 -72) ((p "Oxente_2005_V1_2" 4) -40.6265 -4.73311 -0.251894 -0.0235388 -179 90) ((p "Oxente_2005_V1_2" 5) -37.8752 -22.6443 -0.169217 0.0217794 110 -20) ((p "Oxente_2005_V1_2" 6) -23.4929 -16.6209 -0.00113799 0.000136379 -160 0) ((p "Oxente_2005_V1_2" 7) -24.1104 5.08345 -0.00442364 8.24521e-05 -118 0) ((p "Oxente_2005_V1_2" 8) -25.6176 -6.31744 -0.0245281 -0.00347432 -127 1) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -3.40032e-08 -2.56966e-08 -155 0) ((p "Oxente_2005_V1_2" 10) 4.48956 13.0036 -0.0357388 0.00550099 -141 0) ((p "Oxente_2005_V1_2" 11) 3.42589 -25.08 -0.0529562 -0.00801274 179 -1) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -3.36687e-13 -6.08888e-12 -90 -77) ((p "UvA_Trilearn" 2) 1.43034 -24.5834 -0.00305955 -0.00217932 -157 67) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -1.30301e-07 -3.01817e-08 179 9) ((p "UvA_Trilearn" 4) -12.8507 -5.43843 -0.0804244 -0.0143877 -177 26) ((p "UvA_Trilearn" 5) -3.45103 3.16705 -6.18126e-05 -2.65095e-05 180 36) ((p "UvA_Trilearn" 6) -32.3024 2.27487 -0.69631 -0.109199 -172 68) ((p "UvA_Trilearn" 7) -23.7454 -17.5062 -0.0685581 -0.0103023 171 -19) ((p "UvA_Trilearn" 8) -21.1774 4.75404 -0.0721154 -0.00340049 176 59) ((p "UvA_Trilearn" 9) -32.9481 -10.081 -0.0396015 -0.00098118 -113 -67) ((p "UvA_Trilearn" 10) -38.4935 -21.9036 -0.0893027 0.0831871 -174 -90) ((p "UvA_Trilearn" 11) -33.5269 13.7774 -0.0559928 0.00703973 156 90))

(see_global 74 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -35.6589 -15.4403 0.565265 1.43254) ((p "Oxente_2005_V1_2" 1 goalie) -45.5106 -10.0646 -0.000116155 2.29316e-05 1 -27) ((p "Oxente_2005_V1_2" 2) -40.1028 4.10143 -0.0248936 -0.000287723 99 -89)

((p "Oxente_2005_V1_2" 3) -43.775 -8.87828 0.223516 0.112553 31 -70) ((p "Oxente_2005_V1_2" 4) -40.9865 -4.75816 -0.0411367 -0.00493386 15 -81) ((p "Oxente_2005_V1_2" 5) -38.2327 -21.5048 0.00665515 -0.0204345 110 -39) ((p "Oxente_2005_V1_2" 6) -23.4943 -16.6208 -0.00016362 4.6997e-07 174 0) ((p "Oxente_2005_V1_2" 7) -24.1171 5.08351 -0.00074794 4.4042e-05 -118 0) ((p "Oxente_2005_V1_2" 8) -25.6551 -6.32265 -0.00457809 -0.000874511 73 90) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -5.20828e-09 -4.72349e-09 -161 0) ((p "Oxente_2005_V1_2" 10) 4.43686 13.0079 -0.00606298 0.000340181 -141 0) ((p "Oxente_2005_V1_2" 11) 3.3595 -25.0976 -0.00737449 -0.00200954 167 -1) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -8.52839e-14 -9.28461e-13 -90 -78) ((p "UvA_Trilearn" 2) 1.42524 -24.5863 -0.000778052 -0.000461317 167 -72) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -2.37036e-08 -4.13935e-09 179 7) ((p "UvA_Trilearn" 4) -13.1146 -5.4405 -0.0513944 0.00257188 172 32) ((p "UvA_Trilearn" 5) -3.45111 3.16701 -1.24399e-05 -7.05921e-06 180 31) ((p "UvA_Trilearn" 6) -33.4345 1.9616 -0.215675 -0.0740371 166 90) ((p "UvA_Trilearn" 7) -23.8426 -17.5138 -0.0187766 -0.00218851 177 25) ((p "UvA_Trilearn" 8) -22.0281 4.78517 -0.153597 0.00729084 156 81) ((p "UvA_Trilearn" 9) -33.5428 -11.3898 -0.208682 -0.413215 -113 -5) ((p "UvA_Trilearn" 10) -38.7379 -21.8286 -0.000229267 0.000233526 -179 -1) ((p "UvA_Trilearn" 11) -33.6063 13.7947 -0.0118284 0.00241655 136 90))

(see_global 75 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -35.1284 -14.0487 0.498632 1.30808) ((p "Oxente_2005_V1_2" 1 goalie) -44.9143 -10.0538 0.238497 0.00432756 1 -20) ((p "Oxente_2005_V1_2" 2) -40.2232 4.64965 -0.048144 0.219289 99 -90) ((p "Oxente_2005_V1_2" 3) -43.0893 -8.50602 0.274286 0.148904 31 -67) ((p "Oxente_2005_V1_2" 4) -40.4633 -4.61798 0.209283 0.056072 15 -78) ((p "Oxente_2005_V1_2" 5) -38.2243 -21.5233 -0.000250045 0.000789873 70 -2) ((p "Oxente_2005_V1_2" 6) -23.4945 -16.6208 -6.02385e-05 4.20652e-06 78 80) ((p "Oxente_2005_V1_2" 7) -24.1178 5.08351 -0.000297558 1.733e-06 88 90) ((p "Oxente_2005_V1_2" 8) -25.5273 -5.77807 0.0511206 0.217834 73 90) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -2.22675e-09 -1.87927e-09 -161 0) ((p "Oxente_2005_V1_2" 10) 4.43103 13.0081 -0.00233095 6.8546e-05 -145 0) ((p "Oxente_2005_V1_2" 11) 3.3527 -25.1 -0.00272049 -0.000981144 167 -1) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -3.16521e-14 -3.54751e-13 -90 -80) ((p "UvA_Trilearn" 2) 1.42453 -24.5867 -0.000328196 -0.000194829 145 -72) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -9.7933e-09 -1.76161e-09 179 5) ((p "UvA_Trilearn" 4) -13.1612 -5.44228 -0.0250475 -0.000957391 180 22) ((p "UvA_Trilearn" 5) -3.45112 3.167 -5.60355e-06 -2.71932e-06 180 28) ((p "UvA_Trilearn" 6) -33.6422 1.87734 -0.113999 -0.0462548 159 71) ((p "UvA_Trilearn" 7) -23.8613 -17.5155 -0.0102523 -0.000921281 101 21) ((p "UvA_Trilearn" 8) -22.1731 4.77785 -0.077932 -0.00393233 146 90) ((p "UvA_Trilearn" 9) -33.7731 -11.7696 -0.123739 -0.2041 -150 -71) ((p "UvA_Trilearn" 10) -38.7402 -21.8297 1.11423e-05 -1.12251e-05 -6 31) ((p "UvA_Trilearn" 11) -34.0007 14.1838 -0.183014 0.180531 136 90))

(see_global 80 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -34.1374 -4.18784 0.145261 1.77907) ((p "Oxente_2005_V1_2" 1 goalie) -44.1163 -9.52104 0.15814 0.202456 53 -25) ((p "Oxente_2005_V1_2" 2) -40.6133 6.06342 -0.0037291 0.00963559 89 -90) ((p

"Oxente_2005_V1_2" 3) -42.2289 -6.76514 0.0467801 0.0797629 64 -47) ((p "Oxente_2005_V1_2" 4) -39.6216 -3.00909 0.0915805 0.274111 77 -89) ((p "Oxente_2005_V1_2" 5) -38.0343 -18.346 -0.13646 0.293746 113 -37) ((p "Oxente_2005_V1_2" 6) -23.2933 -14.2067 0.00352393 -0.0287632 101 36) ((p "Oxente_2005_V1_2" 7) -23.9674 7.56932 -0.0337052 0.273619 94 90) ((p "Oxente_2005_V1_2" 8) -24.9196 -3.60033 0.00277188 0.0229481 90 90) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -1.99758e-11 -2.30131e-11 -176 0) ((p "Oxente_2005_V1_2" 10) 4.4272 13.0079 -2.32556e-05 -1.87263e-06 120 82) ((p "Oxente_2005_V1_2" 11) 2.99323 -24.6003 -0.141919 0.200619 121 31) ((p "UvA_Trilearn" 1 goalie) 49.1574 -0.951726 -7.52451e-16 -3.6281e-15 -90 -85) ((p "UvA_Trilearn" 2) 0.353078 -23.3647 -0.103654 0.269572 114 -54) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -9.25383e-11 -2.23794e-12 -180 -2) ((p "UvA_Trilearn" 4) -13.6441 -5.12001 -0.231716 0.172889 138 44) ((p "UvA_Trilearn" 5) -4.07915 3.7593 -0.0993215 0.0916474 123 70) ((p "UvA_Trilearn" 6) -34.0942 0.947629 -0.00083319 -0.0364014 -108 16) ((p "UvA_Trilearn" 7) -23.867 -14.0311 -0.00835949 -0.0689079 81 89) ((p "UvA_Trilearn" 8) -22.6797 5.49043 -0.0768057 0.128717 104 90) ((p "UvA_Trilearn" 9) -34.1117 -12.1026 -0.0126827 0.00525302 107 -16) ((p "UvA_Trilearn" 10) -38.4288 -21.8341 0.05852 0.0158124 20 90) ((p "UvA_Trilearn" 11) -34.709 15.3104 -0.0283889 0.0548708 95 2))

(see_global 82 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -33.8727 -0.916152 0.11555 1.45088) ((p "Oxente_2005_V1_2" 1 goalie) -43.6932 -8.61828 0.108995 0.273958 64 -25) ((p "Oxente_2005_V1_2" 2) -40.7079 7.52529 -0.0160044 0.345732 89 -90) ((p "Oxente_2005_V1_2" 3) -42.217 -6.08815 -0.0148122 0.241552 98 -67) ((p "Oxente_2005_V1_2" 4) -39.1449 -1.19685 0.0844534 0.389024 77 -71) ((p "Oxente_2005_V1_2" 5) -38.386 -17.4086 -0.0875548 0.26629 104 -29) ((p "Oxente_2005_V1_2" 6) -23.3375 -13.6775 -0.0196891 0.222587 94 36) ((p "Oxente_2005_V1_2" 7) -24.174 8.78584 -0.0222326 0.146859 71 90) ((p "Oxente_2005_V1_2" 8) -24.8414 -2.07118 0.0209797 0.339058 90 83) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -3.88446e-12 -4.58194e-12 174 0) ((p "Oxente_2005_V1_2" 10) 4.00737 13.802 -0.0418619 0.0888916 110 90) ((p "Oxente_2005_V1_2" 11) 2.7169 -23.7602 -0.0571286 0.257015 104 43) ((p "UvA_Trilearn" 1 goalie) 49.048 -0.0558044 -0.0197416 0.102097 -90 -88) ((p "UvA_Trilearn" 2) 0.120122 -22.4323 -0.0538857 0.304541 102 43) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -1.45092e-11 -6.32568e-13 -180 -5) ((p "UvA_Trilearn" 4) -14.0239 -4.85261 -0.0707161 0.0455995 105 63) ((p "UvA_Trilearn" 5) -4.78095 4.6511 -0.113221 0.126322 115 74) ((p "UvA_Trilearn" 6) -34.4141 0.233447 -0.0157556 0.00369004 -108 42) ((p "UvA_Trilearn" 7) -23.8718 -14.0907 -4.44863e-05 0.00177612 73 88) ((p "UvA_Trilearn" 8) -22.9696 6.5423 -0.0637566 0.20059 90 90) ((p "UvA_Trilearn" 9) -34.2069 -11.8613 -0.0307198 0.0784395 107 16) ((p "UvA_Trilearn" 10) -38.2297 -21.7761 0.0749176 0.0264165 29 84) ((p "UvA_Trilearn" 11) -34.8726 16.1882 -0.0153982 0.130285 88 1))

(see_global 83 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -33.8292 0.571265 0.0408935 1.39817) ((p "Oxente_2005_V1_2" 1 goalie) -43.3012 -7.77381 0.156812 0.337787 64 -27) ((p "Oxente_2005_V1_2" 2) -40.6745 8.44319 0.0133309 0.36716 89 -90) ((p "Oxente_2005_V1_2" 3) -42.2803 -5.25939 -0.0253459 0.331504 98 -71) ((p "Oxente_2005_V1_2" 4) -39.0797 -0.838005 0.0261063 0.143539 70 -56) ((p

"Oxente_2005_V1_2" 5) -38.6303 -16.49 -0.0977137 0.367436 104 -30) ((p "Oxente_2005_V1_2" 6) -23.4093 -12.8748 0.00412075 -0.0300095 94 35) ((p "Oxente_2005_V1_2" 7) -24.0344 9.55136 0.0558311 0.306208 71 90) ((p "Oxente_2005_V1_2" 8) -24.8571 -1.11296 -0.00630529 0.383288 90 79) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -1.51197e-12 -2.03605e-12 174 0) ((p "Oxente_2005_V1_2" 10) 3.7792 14.4111 -0.0912649 0.243655 110 90) ((p "Oxente_2005_V1_2" 11) 2.63992 -23.4841 -0.0307922 0.110431 127 23) ((p "UvA_Trilearn" 1 goalie) 49.0382 0.0437136 -0.00394508 0.0398072 -90 -88) ((p "UvA_Trilearn" 2) 0.0961116 -22.1172 -0.0111407 0.146186 93 49) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -5.6352e-12 -4.18008e-13 -180 -7) ((p "UvA_Trilearn" 4) -14.2712 -4.23454 -0.132889 0.33215 105 61) ((p "UvA_Trilearn" 5) -5.10174 5.28639 -0.148846 0.294776 115 72) ((p "UvA_Trilearn" 6) -34.4294 0.238163 -0.00840643 0.00258907 -108 11) ((p "UvA_Trilearn" 7) -23.7161 -13.3905 -0.010259 -0.0413228 73 87) ((p "UvA_Trilearn" 8) -22.9872 7.29721 -0.00945932 0.405689 90 90) ((p "UvA_Trilearn" 9) -34.2431 -11.7767 -0.0194876 0.0454581 99 23) ((p "UvA_Trilearn" 10) -38.1541 -21.7447 0.0415484 0.0172502 40 75) ((p "UvA_Trilearn" 11) -34.8868 16.315 -0.00657255 0.0588573 -175 90))

(see_global 86 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -37.0713 0.449909 -2.35211 0.645261) ((p "Oxente_2005_V1_2" 1 goalie) -42.6764 -5.71354 0.0738911 0.372817 75 -37) ((p "Oxente_2005_V1_2" 2) -40.7135 8.99423 -0.00252957 0.020696 -57 -2) ((p "Oxente_2005_V1_2" 3) -42.5986 -2.29952 -0.0188521 0.423589 98 -78) ((p "Oxente_2005_V1_2" 4) -37.7009 -0.178522 0.316436 0.116729 14 -14) ((p "Oxente_2005_V1_2" 5) -38.965 -14.2585 -0.0184674 0.156131 74 0) ((p "Oxente_2005_V1_2" 6) -23.3531 -12.2572 0.000300362 -0.00393411 132 1) ((p "Oxente_2005_V1_2" 7) -23.6819 10.991 0.0113074 0.0575084 -139 2) ((p "Oxente_2005_V1_2" 8) -23.9832 -0.10642 0.0945897 0.0740713 103 82) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -1.08863e-13 -1.30528e-13 174 1) ((p "Oxente_2005_V1_2" 10) 3.24747 15.261 -0.159509 0.20338 134 69) ((p "Oxente_2005_V1_2" 11) 2.31882 -22.7755 -0.109743 0.220425 121 27) ((p "UvA_Trilearn" 1 goalie) 49.0339 0.112212 -9.59051e-05 0.0032283 -90 -89) ((p "UvA_Trilearn" 2) 0.0526882 -21.8587 -0.0045479 0.0153213 131 -90) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -3.87032e-13 -6.89952e-15 -180 -6) ((p "UvA_Trilearn" 4) -14.7417 -2.75356 -0.0382843 0.118892 -161 14) ((p "UvA_Trilearn" 5) -5.71212 6.71188 -0.0288132 0.0917457 -179 8) ((p "UvA_Trilearn" 6) -34.4445 0.243799 -0.00137199 0.000485702 -108 11) ((p "UvA_Trilearn" 7) -23.5433 -12.8616 -0.0023494 -0.0125979 173 -11) ((p "UvA_Trilearn" 8) -22.8873 8.89473 0.0123105 0.121652 118 90) ((p "UvA_Trilearn" 9) -34.3785 -11.1971 -0.0327635 0.115264 99 29) ((p "UvA_Trilearn" 10) -37.7584 -21.4921 0.046384 0.0327009 47 70) ((p "UvA_Trilearn" 11) -34.9095 16.4171 -0.00181485 0.00531283 123 2))

(see_global 87 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -39.3256 1.16663 -2.11905 0.673721) ((p "Oxente_2005_V1_2" 1 goalie) -42.5245 -4.79517 0.0607593 0.367349 75 -40) ((p "Oxente_2005_V1_2" 2) -40.7159 9.01526 -0.000950113 0.0084135 -57 -2) ((p "Oxente_2005_V1_2" 3) -42.6362 -1.88314 -0.0150373 0.166552 71 -29) ((p "Oxente_2005_V1_2" 4) -36.8202 0.0106838 0.352289 0.0756823 14 -21) ((p "Oxente_2005_V1_2" 5) -38.9866 -14.1149 -0.00865728 0.057434 93 0) ((p

"Oxente_2005_V1_2" 6) -23.3528 -12.2608 0.000139083 -0.00145812 131 1) ((p "Oxente_2005_V1_2" 7) -23.6707 11.0473 0.00449482 0.0225017 -148 2) ((p "Oxente_2005_V1_2" 8) -23.89 -0.0228581 0.037288 0.0334249 103 82) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -4.88209e-14 -5.48764e-14 174 1) ((p "Oxente_2005_V1_2" 10) 2.68174 15.9519 -0.226289 0.276387 134 70) ((p "Oxente_2005_V1_2" 11) 2.19155 -22.5689 -0.0509099 0.0826345 121 27) ((p "UvA_Trilearn" 1 goalie) 49.0337 0.115339 -7.42766e-05 0.00125076 -90 -90) ((p "UvA_Trilearn" 2) -0.306662 -21.4284 -0.166739 0.199671 131 -70) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -1.63112e-13 5.66335e-15 -180 -6) ((p "UvA_Trilearn" 4) -14.7885 -2.63197 -0.0251652 0.0653425 -172 -14) ((p "UvA_Trilearn" 5) -5.73476 6.79461 -0.0105047 0.0383847 -174 5) ((p "UvA_Trilearn" 6) -34.4458 0.244317 -0.000709274 0.000284389 170 -68) ((p "UvA_Trilearn" 7) -23.5444 -12.8741 -0.000593307 -0.00690921 126 -25) ((p "UvA_Trilearn" 8) -22.8635 9.01976 0.0127661 0.0671896 137 34) ((p "UvA_Trilearn" 9) -34.4083 -11.0926 -0.0160229 0.0561861 118 -40) ((p "UvA_Trilearn" 10) -37.7067 -21.4612 0.0284061 0.0169433 121 -71) ((p "UvA_Trilearn" 11) -35.1844 16.9011 -0.127592 0.224556 123 90))

(see_global 89 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -43.5585 2.54774 -1.94149 0.613319) ((p "Oxente_2005_V1_2" 1 goalie) -42.1053 -3.04206 0.0907056 0.338687 75 29) ((p "Oxente_2005_V1_2" 2) -40.7165 9.02683 -2.50088e-06 0.00130349 174 73) ((p "Oxente_2005_V1_2" 3) -42.6697 -1.64992 -0.00270692 0.0263704 136 -32) ((p "Oxente_2005_V1_2" 4) -36.3997 -0.028391 -0.00475962 -0.000792979 173 -12) ((p "Oxente_2005_V1_2" 5) -38.9966 -14.0348 -0.000559015 0.00883532 125 -19) ((p "Oxente_2005_V1_2" 6) -23.3527 -12.2628 2.90821e-06 -0.000225823 143 1) ((p "Oxente_2005_V1_2" 7) -23.6674 11.0807 0.000136522 0.0035956 -156 1) ((p "Oxente_2005_V1_2" 8) -23.8331 0.0273538 0.00633684 0.00510832 -106 1) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -7.93465e-15 -8.02828e-15 173 1) ((p "Oxente_2005_V1_2" 10) 2.33571 16.3053 -0.0422682 0.0383297 -163 1) ((p "Oxente_2005_V1_2" 11) 2.10927 -22.4552 -0.0103657 0.0127799 151 0) ((p "UvA_Trilearn" 1 goalie) 49.0336 0.116952 -2.38574e-05 0.000172855 84 87) ((p "UvA_Trilearn" 2) -0.906113 -20.7145 0.000680363 0.00267515 131 20) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -2.67589e-14 3.26535e-15 -180 -7) ((p "UvA_Trilearn" 4) -14.8333 -2.52865 -0.00861741 0.0202596 156 15) ((p "UvA_Trilearn" 5) -5.74764 6.85072 -0.00138196 0.0077797 -179 6) ((p "UvA_Trilearn" 6) -36.2854 0.56062 0.0628296 -0.00407393 170 15) ((p "UvA_Trilearn" 7) -24.1217 -12.3929 -0.316465 0.268208 144 29) ((p "UvA_Trilearn" 8) -23.3176 9.39526 -0.249768 0.163358 147 87) ((p "UvA_Trilearn" 9) -34.8032 -10.5393 -0.206375 0.266565 128 29) ((p "UvA_Trilearn" 10) -37.6652 -21.4342 0.00777492 0.00537583 109 27) ((p "UvA_Trilearn" 11) -35.4083 17.2045 -0.0327366 0.0416164 144 90))

(see_global 90 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -45.5711 3.08593 -1.8918 0.505897) ((p "Oxente_2005_V1_2" 1 goalie) -41.8534 -2.21083 0.10074 0.332494 75 49) ((p "Oxente_2005_V1_2" 2) -41.2648 9.05728 -0.219306 0.01218 174 62) ((p "Oxente_2005_V1_2" 3) -43.0955 -1.25778 -0.170322 0.156858 136 -18) ((p "Oxente_2005_V1_2" 4) -36.4046 -0.0294147 -0.00194091 -0.00040945 101 -13) ((p "Oxente_2005_V1_2" 5) -38.9975 -14.0265 -0.000344007 0.00330962 134 -23) ((p

"Oxente_2005_V1_2" 6) -23.8409 -11.876 -0.195307 0.154693 143 2) ((p "Oxente_2005_V1_2" 7) -23.6676 11.084 -7.00325e-05 0.00130871 170 29) ((p "Oxente_2005_V1_2" 8) -23.826 0.0318728 0.00281538 0.00180761 163 10) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -3.15219e-15 -3.35218e-15 172 1) ((p "Oxente_2005_V1_2" 10) 2.29182 16.3399 -0.0175555 0.0138575 -168 1) ((p "Oxente_2005_V1_2" 11) 2.09756 -22.443 -0.00468339 0.00487763 151 0) ((p "UvA_Trilearn" 1 goalie) 49.0335 0.117121 -5.62035e-06 6.75923e-05 84 90) ((p "UvA_Trilearn" 2) -0.905698 -20.7121 0.000192221 0.00112805 117 -55) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -1.05957e-14 1.05166e-15 -180 -7) ((p "UvA_Trilearn" 4) -15.3694 -2.29287 -0.288065 0.126707 156 14) ((p "UvA_Trilearn" 5) -5.7491 6.85893 -0.000674961 0.00381216 -179 5) ((p "UvA_Trilearn" 6) -36.2208 0.554645 0.0354475 -0.00328048 165 -76) ((p "UvA_Trilearn" 7) -24.4351 -12.1105 -0.172008 0.155081 158 -13) ((p "UvA_Trilearn" 8) -23.5541 9.57235 -0.127072 0.0951665 161 2) ((p "UvA_Trilearn" 9) -34.9877 -10.2595 -0.0991301 0.15039 139 -46) ((p "UvA_Trilearn" 10) -37.6569 -21.4297 0.00456734 0.00244173 127 -54) ((p "UvA_Trilearn" 11) -35.9617 17.5651 -0.256761 0.167344 144 57))

(see_global 93 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -50.9999 4.94126 -1.6418 0.5708) ((p "Oxente_2005_V1_2" 1 goalie) -41.4635 -0.701506 0.0197992 0.0594153 103 47) ((p "Oxente_2005_V1_2" 2) -42.3852 9.3778 -0.0991432 0.0317117 149 61) ((p "Oxente_2005_V1_2" 3) -43.9928 -0.68523 -0.0847217 0.0408502 143 -1) ((p "Oxente_2005_V1_2" 4) -37.7758 0.12317 -0.303432 0.0414828 174 -13) ((p "Oxente_2005_V1_2" 5) -40.0412 -12.9745 -0.192544 0.161135 141 -20) ((p "Oxente_2005_V1_2" 6) -24.8332 -11.3492 -0.0873261 0.0409511 169 -19) ((p "Oxente_2005_V1_2" 7) -25.8848 11.6698 -0.335029 0.0941476 170 26) ((p "Oxente_2005_V1_2" 8) -25.9879 0.782479 -0.330372 0.130195 163 8) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -1.87204e-16 -2.3326e-16 171 1) ((p "Oxente_2005_V1_2" 10) 2.26232 16.3588 -0.00133614 0.0006879 -170 1) ((p "Oxente_2005_V1_2" 11) 2.08923 -22.4348 -0.00034209 0.000350063 152 0) ((p "UvA_Trilearn" 1 goalie) 49.0335 0.117235 -1.69691e-07 4.86912e-06 84 90) ((p "UvA_Trilearn" 2) -1.19281 -20.1384 -0.000176178 0.00133467 156 84) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -6.60687e-16 1.51493e-16 -180 -7) ((p "UvA_Trilearn" 4) -16.8746 -1.55466 -0.132087 0.0578011 -169 -21) ((p "UvA_Trilearn" 5) -5.75072 6.86532 -0.000136072 0.000434563 -180 3) ((p "UvA_Trilearn" 6) -39.0032 1.42776 -0.651359 0.207595 165 -82) ((p "UvA_Trilearn" 7) -26.7428 -11.2541 -0.55585 0.17963 165 -18) ((p "UvA_Trilearn" 8) -26.1166 10.4976 -0.54182 0.18458 161 33) ((p "UvA_Trilearn" 9) -36.3003 -9.32673 -0.391253 0.250035 147 -46) ((p "UvA_Trilearn" 10) -37.7732 -21.305 -0.022098 0.0237201 146 -65) ((p "UvA_Trilearn" 11) -37.0545 17.8135 -0.31368 0.00391612 179 9))

(see_global 94 ((g l) -52.5 0) ((g r) 52.5 0) ((b) -52.6066 5.58628 0 0) ((p "Oxente_2005_V1_2" 1 goalie) -41.4434 -0.647953 0.00800985 0.0214214 123 29) ((p "Oxente_2005_V1_2" 2) -43.0167 9.75986 -0.252617 0.152824 149 56) ((p "Oxente_2005_V1_2" 3) -44.6201 -0.244655 -0.250926 0.17623 143 1) ((p "Oxente_2005_V1_2" 4) -38.6473 0.172354 -0.348611 0.0196737 174 -13) ((p "Oxente_2005_V1_2" 5) -40.6754 -12.4861 -0.253674 0.195392 141 -17) ((p "Oxente_2005_V1_2" 6) -25.5338 -11.228 -0.28023 0.0485112 169 -19) ((p

"Oxente_2005_V1_2" 7) -26.9014 11.9481 -0.406619 0.111305 170 24) ((p "Oxente_2005_V1_2" 8) -26.3165 0.891723 -0.131439 0.0436975 169 1) ((p "Oxente_2005_V1_2" 9) 1.12329 -3.44529 -7.75376e-17 -1.01846e-16 170 1) ((p "Oxente_2005_V1_2" 10) 2.2611 16.3595 -0.000488288 0.000261214 -169 1) ((p "Oxente_2005_V1_2" 11) 2.08884 -22.4344 -0.000155827 0.000140691 153 0) ((p "UvA_Trilearn" 1 goalie) 49.0335 0.117239 8.02346e-08 1.89386e-06 84 90) ((p "UvA_Trilearn" 2) -1.37614 -20.0423 -0.0850652 0.0445827 156 84) ((p "UvA_Trilearn" 3) 3.29422 -3.69422 -2.61618e-16 4.95891e-17 -180 -7) ((p "UvA_Trilearn" 4) -17.0169 -1.50866 -0.0764691 0.0247243 -175 -15) ((p "UvA_Trilearn" 5) -5.75084 6.86575 -5.52349e-05 0.000196355 -180 3) ((p "UvA_Trilearn" 6) -40.1145 1.88887 -0.610107 0.253149 165 78) ((p "UvA_Trilearn" 7) -27.7853 -10.9547 -0.572322 0.164397 165 -18) ((p "UvA_Trilearn" 8) -27.2794 10.798 -0.62487 0.161444 161 33) ((p "UvA_Trilearn" 9) -37.2109 -8.87233 -0.489327 0.244194 147 25) ((p "UvA_Trilearn" 10) -38.3843 -20.8905 -0.335498 0.227559 146 6) ((p "UvA_Trilearn" 11) -37.8802 17.8577 -0.383136 0.0205249 179 75))

(see_global 96 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 0 0 0 0) ((p "Oxente_2005_V1_2" 1 goalie) -44.9851 0.0348427 2.97502e-24 1.31685e-23 0 1) ((p "Oxente_2005_V1_2" 2) -16.1898 16.0766 -5.49156e-24 3.12885e-22 -45 0) ((p "Oxente_2005_V1_2" 3) -17.3958 5.30717 -2.66992e-22 1.29395e-22 -16 0) ((p "Oxente_2005_V1_2" 4) -17.2032 -4.9843 -2.20455e-22 9.15839e-23 17 -1) ((p "Oxente_2005_V1_2" 5) -16.4236 -15.7018 -3.23166e-22 -1.56435e-23 43 0) ((p "Oxente_2005_V1_2" 6) -8.16903 0.0566694 -1.81975e-22 2.01356e-22 0 0) ((p "Oxente_2005_V1_2" 7) -5.63579 10.1062 -2.70457e-22 1.93676e-22 -63 1) ((p "Oxente_2005_V1_2" 8) -5.19008 -9.92348 -8.92154e-23 1.16276e-22 63 0) ((p "Oxente_2005_V1_2" 9) -0.640046 0.000705181 0.313745 -0.0053649 -1 0) ((p "Oxente_2005_V1_2" 10) -2.0009 22.0005 -6.94214e-25 1.85354e-25 -86 1) ((p "Oxente_2005_V1_2" 11) -2.0001 -21.9999 -1.11427e-25 1.25364e-25 85 0) ((p "UvA_Trilearn" 1 goalie) 49.0335 0.117243 6.49939e-28 6.63559e-28 -175 -4) ((p "UvA_Trilearn" 2) 15.8558 -9.91971 -2.18631e-19 -1.12291e-19 150 -1) ((p "UvA_Trilearn" 3) 21 5.88164e-18 -2.23629e-37 -1.20177e-38 -180 0) ((p "UvA_Trilearn" 4) 14.9998 0.000119228 -4.69337e-18 3.59032e-18 -179 -1) ((p "UvA_Trilearn" 5) 16 10 -1.12521e-22 2.80484e-22 -149 0) ((p "UvA_Trilearn" 6) 1.12247 11.195 -5.6401e-15 4.69049e-15 -98 1) ((p "UvA_Trilearn" 7) 1.73671 -10.8819 -8.62492e-15 9.30337e-15 100 0) ((p "UvA_Trilearn" 8) 7.82826 4.74186 0 0 -150 1) ((p "UvA_Trilearn" 9) 8.12213 4.21876 0 0 -153 0) ((p "UvA_Trilearn" 10) 1.90359 -18.8976 -3.91748e-15 9.44717e-15 95 0) ((p "UvA_Trilearn" 11) 1.33066 19.0247 -1.48525e-18 2.61102e-19 -94 0))

(see_global 99 ((g l) -52.5 0) ((g r) 52.5 0) ((b) 7.09211 -0.759782 2.09101 -0.237719) ((p "Oxente_2005_V1_2" 1 goalie) -43.6353 0.00233254 0.313548 0.000302238 0 1) ((p "Oxente_2005_V1_2" 2) -16.1898 16.0766 7.1179e-25 1.52083e-23 -64 30) ((p "Oxente_2005_V1_2" 3) -17.3958 5.30717 -1.34683e-23 6.68935e-24 -103 87) ((p "Oxente_2005_V1_2" 4) -17.2032 -4.9843 -1.48813e-23 5.84685e-24 38 -32) ((p "Oxente_2005_V1_2" 5) -16.4236 -15.7018 -2.34672e-23 7.14154e-25 58 -24) ((p "Oxente_2005_V1_2" 6) -7.79819 -0.349672 0.148336 -0.162536 -44 46) ((p "Oxente_2005_V1_2" 7) -5.07703 10.0739 0.223502 -

0.0129214 1 -44) ((p "Oxente_2005_V1_2" 8) -5.00262 -9.331 0.0749846 0.236993 72 -32) ((p "Oxente_2005_V1_2" 9) 0.432729 -0.103999 0.255667 -0.0353382 -6 1) ((p "Oxente_2005_V1_2" 10) -1.46671 21.95 0.213675 -0.0201782 -8 -62) ((p "Oxente_2005_V1_2" 11) -1.43971 -21.9278 0.224153 0.0288484 8 60) ((p "UvA_Trilearn" 1 goalie) 49.0335 0.117243 3.11159e-29 4.20342e-29 -90 -89) ((p "UvA_Trilearn" 2) 15.8558 -9.91971 -2.55361e-20 -8.70177e-21 15 90) ((p "UvA_Trilearn" 3) 21 5.88164e-18 -1.48155e-38 2.26436e-40 -3 -90) ((p "UvA_Trilearn" 4) 14.9998 0.000119228 -6.16762e-19 6.756e-19 -144 -90) ((p "UvA_Trilearn" 5) 16 10 -1.38146e-23 2.29328e-23 -16 -90) ((p "UvA_Trilearn" 6) 1.12247 11.195 -9.83832e-16 9.54206e-16 -6 -90) ((p "UvA_Trilearn" 7) 1.73671 -10.8819 -1.22902e-15 1.35325e-15 -176 -90) ((p "UvA_Trilearn" 8) 7.82826 4.74186 0 0 163 0) ((p "UvA_Trilearn" 9) 8.12213 4.21876 0 0 -169 32) ((p "UvA_Trilearn" 10) 1.23657 -18.9586 -0.366193 -0.0334879 -180 -86) ((p "UvA_Trilearn" 11) 0.700275 19.0918 -0.292497 0.0311643 173 88))

APÊNDICE II

Para implementação do método, se fez necessária a definição de sete classes. Sendo estas: Ponto2D, Reta, Circulo, Inf_Frame, Historico, RdP, Construtor_RdP.

A classe Ponto2D foi definida com a finalidade de representar as localizações dos objetos (bola e jogadores) no campo. Possuindo os seguintes métodos:

- Ponto2D() – construtor que cria um ponto com coordenadas x e y nulas;
- Ponto2D(double v_x, double v_y) – construtor que cria um ponto com coordenadas x e y recebidas como entrada;
- void setar_xy(double v_x, double v_y) – recebe como entrada as coordenadas que serão setadas como coordenadas do objeto receptor da mensagem que invoca tal método;
- double consultar_x() – retorna a coordenada x do objeto receptor da mensagem que invoca tal método;
- double consultar_y() – retorna a coordenada y do objeto receptor da mensagem que invoca tal método;
- double distancia_ao_ponto(Ponto2D p) – recebe como entrada um objeto da classe Ponto2D e retorna a distância entre o ponto recebido como entrada e o representado pelo objeto receptor da mensagem que invoca tal método.

A classe Reta, por sua vez, foi definida para possibilitar a construção do método que implementará o Classificador. Possuindo os seguintes métodos:

- Reta(Ponto2D v_p1, Ponto2D v_p2) – construtor que cria uma reta com base nos dois pontos recebidos como entrada;
- double consultar_x_p1 () – retorna a coordenada x do ponto p1, membro do objeto receptor da mensagem que invoca tal método;

- double consultar_y_p1 () – retorna a coordenada y do ponto p1, membro do objeto receptor da mensagem que invoca tal método;
- double consultar_x_p2 () – retorna a coordenada x do ponto p2, membro do objeto receptor da mensagem que invoca tal método;
- double consultar_y_p2 () – retorna a coordenada y do ponto p2, membro do objeto receptor da mensagem que invoca tal método;
- Ponto2D projecao_do_ponto (Ponto2D p) – retorna o ponto que representa a projeção do ponto, recebido como entrada, sobre a reta imagem do objeto receptor da mensagem que invoca tal método;
- double distancia_ao_ponto (Ponto2D p) – retorna a distância do ponto, recebido como entrada, à reta imagem do objeto receptor da mensagem que invoca tal método;
- Ponto2D ponto_de_interseccao_com_a_reta (Reta r) – retorna o ponto de intersecção entre a reta, recebida como entrada, e a reta imagem do objeto receptor da mensagem que invoca tal método.

A classe Circulo assim com a classe Reta foi definida para possibilitar a construção do método que implementará o Classificador. Possuindo os seguintes métodos:

- Circulo(double r) – construtor que cria um circulo com centro em x e y iguais a zero e raio r;
- Circulo(Ponto2D c, double r) – construtor que cria um circulo com centro em c e raio r;
- bool ponto_pertence_ao_circulo (Ponto2D p) – retorna um lógico representando a pertinência do ponto, recebido como entrada, no circulo representado pelo objeto receptor da mensagem que invoca tal método.

A classe Inf_Frame foi definida visando que os objetos pertencentes a esta representem estados de jogo, ou seja, armazenem as posições dos jogadores e da

bola, além de qual time possui a posse de bola e que jogador deste time a detêm em um determinado instante da partida. Possuindo os seguintes métodos:

- void setar_coordenadas_jogador (bool oponente , int jogador, double x, double y) – seta, com os valores dos parâmetros formais x e y, as coordenadas de um determinado jogador, indicado através do número do jogador contido no parâmetro formal jogador e do valor lógico, do parâmetro oponente, o qual indica se o jogador em questão é do time oponente;

- void setar_coordenadas_bola (double x, double y) – seta as coordenadas da bola com os valores contidos nos parâmetros formais x e y;

- void setar_bola_com_oponente(bool posse) – seta com o valor do parâmetro formal posse o membro, que identifica se a bola está em posse do time oponente, pertencente ao objeto receptor da mensagem que invoca tal método;

- void setar_jogador_com_posse_da_bola(int jogador) – seta com o valor do parâmetro formal jogador o membro, que identifica o número do jogador que possui a posse de bola, pertencente ao objeto receptor da mensagem que invoca tal método;

- double coordenada_x_bola () – retorna a coordena x da bola no quadro representado pelo objeto receptor da mensagem que invoca tal método;

- double coordenada_y_bola () – retorna a coordena y da bola no quadro representado pelo objeto receptor da mensagem que invoca tal método;

- Ponto2D consultar_coordenadas_bola () – retorna o ponto que representa a localização da bola no quadro representado pelo objeto receptor da mensagem que invoca tal método;

- Ponto2D consultar_coordenadas_jogador (bool oponente , int jogador) – retorna o ponto que representa a localização do jogador, determinado pelos

valores recebidos como entrada, no quadro representado pelo objeto receptor da mensagem que invoca tal método;

- int consultar_jogador_com_posse_da_bola () – retorna o número do jogador que possui a posse de bola no quadro representado pelo objeto receptor da mensagem que invoca tal método;

- bool posse_bola_oponente () – retorna verdadeiro caso o time oponente possua a posse de bola no quadro representado pelo objeto receptor da mensagem que invoca tal método, retornando falso caso contrário;

- int retorna_quadrante_bola() – retorna o valor do quadrante em que a bola se encontra no quadro representado pelo objeto receptor da mensagem que invoca tal método;

- int retorna_quadrante_bola_fora_lateral() – retorna o valor do quadrante em que a bola se encontra no quadro representado pelo objeto receptor da mensagem que invoca tal método, após uma ocorrência de saída de bola na lateral do campo;

- int retorna_quadrante_bola_fora_fundo() – retorna o valor do quadrante em que a bola se encontra no quadro representado pelo objeto receptor da mensagem que invoca tal método, após uma ocorrência de saída de bola no fundo do campo;

- int jogador_mais_proximo_da_bola (bool oponente) – retorna o número do jogador oponente mais próximo da bola no quadro representado pelo objeto receptor da mensagem que invoca tal método, quando o valor do parâmetro formal oponente for verdadeiro, caso contrário, retorna o número do jogador que não pertence ao time oponente mais próximo da bola no quadro representado pelo objeto receptor da mensagem que invoca tal método.

A classe Historico, por sua vez, foi definida para implementar através de um de seus métodos o Classificador, os demais métodos são utilizados para dar suporte a

implementação do Classificador ou do módulo principal. A classe Historico possui os seguintes métodos:

- bool bola_mantem_direcao_e_sentido() – retorna verdadeiro caso a bola mantenha a mesma direção e sentido em sua trajetória, retorna falso caso contrario;

- bool oponente_esta_no_campo_esquerdo() – retorna verdadeiro caso o time oponente efetue gol´s na trave posicionada no campo direito, retorna falso caso contrário;

- void inicializar_historico(char *nome) – inicializa o membro nome_time com o valor do parâmetro formal nome;

- int atualizar_historico(char *visao_global) – implementa o modulo Classificador do MOMCoTO;

- Inf_Frame retorna_frame(int n) – retorna um dos quatro quadros utilizados para armazenar o estado interno do objeto receptor da mensagem que invoca tal método, sendo o quadro a ser retornado determinado pelo valor de n que pertence ao intervalo [1, 4];

- void restaurar_historico() – restaura o estado do histórico para que este reflita o estado imediatamente anterior ao desvio na trajetória da bola;

- bool atualizar_informacoes_para_analise_da_ocorrencia_de_conducao() – verifica se o jogador que possuía a posse de bola antes do desvio é o mesmo jogador que possui a posse de bola após o desvio, caso positivo as informações necessárias para a verificação da ocorrência de condução são ajustadas, caso negativo nada é feito. O resultado da verificação é retornado;

- int consultar_ciclo() – retorna o valor do ciclo referente à última visão global utilizada para atualizar o histórico.

A classe RdP, foi definida para que um objeto instanciado desta classe venha a representar o modelo comportamental do time oponente. Como foi mencionado

anteriormente, foi estabelecida a forma matricial para se representar a RdP. No entanto, alguns detalhes da implementação devem ser mencionados, para se obter uma melhor utilização da memória no armazenamento da RdP e também uma eficiência futura na utilização do modelo comportamental gerado, foram utilizadas matrizes esparsas para representar a RdP e por sua vez foram utilizadas listas cruzadas como estruturas de dados na implementação das matrizes esparsas. A classe RdP possui os seguintes métodos:

- RdP() – construtor que inicializa as listas cruzadas utilizadas para armazenar a RdP;

- ~RdP() – destrutor responsável pela desalocação da memória utilizada para armazenar a RdP;

- int localizar_posicao_de_insercao_lugar(char desc[7]) – retorna a posição de inserção do lugar cuja descrição é recebida como entrada, mais especificamente, retorna a posição em que o lugar será inserido na lista que armazena os endereços dos primeiros elementos das colunas. Caso o lugar já esteja presente na lista é retornado o inteiro zero;

- void inserir_lugar(Inf_Frame frame, Situacao s) – insere o lugar descrito através dos parâmetros formais na posição adequada nas matrizes que representam a RdP em questão;

- int localizar_posicao_de_insercao_transicao(Transicao desc, char ld[7]) – retorna a posição de inserção da transição cuja descrição é recebida como entrada, mais especificamente, retorna a posição em que a transição será inserida na lista que armazena os endereços dos primeiros elementos das linhas. Caso a transição já esteja presente na lista é retornado o inteiro zero;

- void inserir_transicao(Transicao desc, Inf_Frame frame_LO, Inf_Frame frame_LD, Situacao s) – insere a transição descrita através dos parâmetros formais na posição adequada nas matrizes que representam a RdP em questão;

- int localizar_indice_do_lugar(char desc[7]) – retorna a posição em que o lugar, cuja descrição é recebida como entrada, encontra-se na lista que armazena os endereços dos primeiros elementos das colunas;

- int localizar_indice_da_transicao(Transicao desc, char ld[7]) – retorna a posição em que a transição, cuja descrição é recebida como entrada, encontra-se na lista que armazena os endereços dos primeiros elementos das linhas;

- void escrever_RdP_em_arquivo (FILE *pf) – escreve as matrizes que representam a RdP no arquivo texto cujo ponteiro para este é recebido como entrada.

A classe Construtor_RdP, foi definida para implementar o módulo principal do método proposto. Possuindo os seguintes métodos:

- Construtor_RdP() – construtor que efetua a inicialização dos membros utilizados nas manipulações efetuadas pelo módulo principal implementado através do método atualizar_RdP();

- ~Construtor_RdP() – destrutor que fecha o arquivo onde a RdP foi escrita;

- void atualizar_RdP(RdP &modelo_do_oponente, char *vg) – implementa o módulo principal do método proposto.

A seguir é apresentado o código do arquivo MOMCoTO.h, o qual contém a descrição de cada uma das classes, anteriormente mencionadas, assim como as demais definições que se fizeram necessárias para a implementação das mesmas.

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <stdlib.h>
#define tolerancia      0.09
#define p1x             -47.0
#define p1y             9.0
#define p2x             -36.0
#define p2y             -20.0
```

```
#define p3x                0.0
#define p3y                0.0
#define p4x                20.0
#define p4y                -9.0
#define p5x                30.0
#define p5y                34.0
#define p6x                52.5
#define p6y                -34
#define p7x                52.5
#define p7y                7.01
enum Situacao
{
   gol,
   bola_fora_l,
   bola_fora_f,
   bola_em_jogo
};
enum Transicao {
   passe,
   chute,
   conducao,
   perda_posse_bola
};
enum Estado {
   pre_inicial,
   inicial,
   inicio_2_tempo,
   pos_gol,
   pos_inicial,
   pos_saida_de_bola_lateral,
   pos_saida_de_bola_fundo,
   corrente,
   analise
};
class Ponto2D
{
   double x;
   double y;
public:
   Ponto2D();
   Ponto2D(double v_x, double v_y);
   ~Ponto2D();
   void setar_xy(double v_x, double v_y);
   double consultar_x();
   double consultar_y();
   double distancia_ao_ponto(Ponto2D p);
};
Ponto2D::Ponto2D()
```

```
{
  x=0.0;
  y=0.0;
}
Ponto2D::Ponto2D(double v_x, double v_y)
{
  x=v_x;
  y=v_y;
}
Ponto2D::~Ponto2D()
{
}
void Ponto2D::setar_xy(double v_x, double v_y)
{
  x=v_x;
  y=v_y;
}
double Ponto2D::consultar_x()
{
  return x;
}
double Ponto2D::consultar_y()
{
  return y;
}
double Ponto2D::distancia_ao_ponto(Ponto2D p)
{
  return sqrt(pow(p.consultar_y()-y,2)+pow(p.consultar_x()-x,2));
}
class Reta
{
  Ponto2D p1;
  Ponto2D p2;
public:
  Reta(Ponto2D v_p1, Ponto2D v_p2);
  ~Reta();
  double consultar_x_p1 ();
  double consultar_y_p1 ();
  double consultar_x_p2 ();
  double consultar_y_p2 ();
  Ponto2D projecao_do_ponto (Ponto2D p);
  double distancia_ao_ponto (Ponto2D p);
  Ponto2D ponto_de_interseccao_com_a_reta (Reta r);
};
Reta::Reta(Ponto2D v_p1, Ponto2D v_p2)
{
  p1=v_p1;
  p2=v_p2;
```

```
}
Reta::~Reta()
{
}
double Reta::consultar_x_p1 ()
{
  return p1.consultar_x();
}
double Reta::consultar_y_p1 ()
{
  return p1.consultar_y();
}
double Reta::consultar_x_p2 ()
{
  return p2.consultar_x();
}
double Reta::consultar_y_p2 ()
{
  return p2.consultar_y();
}
Ponto2D Reta::projecao_do_ponto (Ponto2D p)
{
  double x=p.consultar_x();
  double y=p.consultar_y();
  double Ox=p1.consultar_x();
  double Oy=p1.consultar_y();
  double Dx=p2.consultar_x();
  double Dy=p2.consultar_y();
  double y_proj = (x + y*(Dy-Oy)/(Dx-Ox) + Oy*(Dx-Ox)/(Dy-Oy) - Ox) * (Dy-Oy) *
(Dx-Ox) / ((Dx-Ox)*(Dx-Ox) + (Dy-Oy)*(Dy-Oy));
  double x_proj = x + (Dy-Oy)*(y-(y_proj))/(Dx-Ox);
  Ponto2D ponto(x_proj, y_proj);
  return ponto;
}
double Reta::distancia_ao_ponto (Ponto2D p)
{
  return p.distancia_ao_ponto(projecao_do_ponto (p));
}
Ponto2D Reta::ponto_de_interseccao_com_a_reta (Reta r)
{
  double x1=r.consultar_x_p1();
  double y1=r.consultar_y_p1();
  double x2=r.consultar_x_p2();
  double y2=r.consultar_y_p2();
  double x3=consultar_x_p1();
  double y3=consultar_y_p1();
  double x4=consultar_x_p2();
  double y4=consultar_y_p2();
```

```
   double x=(y3-x3*(y3-y4)/(x3-x4)+x1*(y2-y1)/(x2-x1)-y1)/((y2-y1)/(x2-x1)-(y3-y4)/(x3-
x4));
   double y=(x-x1)*(y2-y1)/(x2-x1)+y1;
   Ponto2D p(x,y);
   return p;
}
class Circulo
{
   Ponto2D centro;
   double raio;
public:
   Circulo(double r);
   Circulo(Ponto2D c, double r);
   bool ponto_pertence_ao_circulo (Ponto2D p);
};
Circulo::Circulo(double r)
{
   Ponto2D c;
   centro=c;
   raio=r;
}
Circulo::Circulo(Ponto2D c, double r)
{
   centro=c;
   raio=r;
}
bool Circulo::ponto_pertence_ao_circulo (Ponto2D p)
{
   return (centro.distancia_ao_ponto(p)<=raio?true:false);
}
class Inf_Frame
{
   Ponto2D coordenadas_time_oponente[11];
   Ponto2D coordenadas_time[11];
   Ponto2D coordenadas_bola;
   bool bola_com_oponente;
   int jogador_com_posse_da_bola;
 public:
   void setar_coordenadas_jogador (bool oponente , int jogador, double x, double y);
   void setar_coordenadas_bola (double x, double y);
   void setar_bola_com_oponente(bool posse);
   void setar_jogador_com_posse_da_bola(int jogador);
   double coordenada_x_bola ();
   double coordenada_y_bola ();
   Ponto2D consultar_coordenadas_bola ();
   Ponto2D consultar_coordenadas_jogador (bool oponente , int jogador);
   int consultar_jogador_com_posse_da_bola ();
   bool posse_bola_oponente ();
```

```
   int retorna_quadrante_bola();
   int retorna_quadrante_bola_fora_lateral();
   int retorna_quadrante_bola_fora_fundo();
   int jogador_mais_proximo_da_bola (bool oponente);
};
int Inf_Frame::retorna_quadrante_bola()
{
   double x=coordenada_x_bola();
   double y=coordenada_y_bola();
   if (p6y>y)
   {
      if (-p6x<=x && x<p1x)
         return 1;
      if (p1x<=x && x<p2x)
         return 2;
      if (p2x<=x && x<-p5x)
         return 3;
      if (-p5x<=x && x<-p4x)
         return 4;
      if (-p4x<=x && x<p4y)
         return 5;
      if (p4y<=x && x<p3x)
         return 6;
      if (p3x<=x && x<p1y)
         return 7;
      if (p1y<=x && x<p4x)
         return 8;
      if (p4x<=x && x<p5x)
         return 9;
      if (p5x<=x && x<-p2x)
         return 10;
      if (-p2x<=x && x<-p1x)
         return 11;
      if (-p1x<=x && x<=p6x)
         return 12;
   }
   if (p2y>y && -p6x>x)
      return 13;
   if (p6y<=y && y<p2y)
   {
      if (-p6x<=x && x<p1x)
         return 14;
      if (p1x<=x && x<p2x)
         return 15;
      if (p2x<=x && x<-p5x)
         return 16;
      if (-p5x<=x && x<-p4x)
         return 17;
```

```
    if (-p4x<=x && x<p4y)
      return 18;
    if (p4y<=x && x<p3x)
      return 19;
    if (p3x<=x && x<p1y)
      return 20;
    if (p1y<=x && x<p4x)
      return 21;
    if (p4x<=x && x<p5x)
      return 22;
    if (p5x<=x && x<-p2x)
      return 23;
    if (-p2x<=x && x<-p1x)
      return 24;
    if (-p1x<=x && x<=p6x)
      return 25;
  }
  if (p2y>y && p6x<x)
    return 26;
  if (p2y<=y && y<p4y)
  {
    if (-p6x>x)
      return 27;
    if (-p6x<=x && x<p1x)
      return 28;
    if (p1x<=x && x<p2x)
      return 29;
    if (p2x<=x && x<-p5x)
      return 30;
    if (-p5x<=x && x<-p4x)
      return 31;
    if (-p4x<=x && x<p4y)
      return 32;
    if (p4y<=x && x<p3x)
      return 33;
    if (p3x<=x && x<p1y)
      return 34;
    if (p1y<=x && x<p4x)
      return 35;
    if (p4x<=x && x<p5x)
      return 36;
    if (p5x<=x && x<-p2x)
      return 37;
    if (-p2x<=x && x<-p1x)
      return 38;
    if (-p1x<=x && x<=p6x)
      return 39;
    if (x>p6x)
```

```
      return 40;
  }
  if (-p6x>x)
  {
    if (p4y<y && y<-p7y)
      return 41;
    if (-p7y<=y && y<p3y)
      return 55;
    if (p3y<=y && y<=p7y)
      return 57;
    if (p7y<y && y<p1y)
      return 59;
  }
  if (p4y<=y && y<p1y)
  {
    if (-p6x<=x && x<p1x)
      return 42;
    if (p1x<=x && x<p2x)
      return 43;
    if (p2x<=x && x<-p5x)
      return 44;
    if (-p5x<=x && x<-p4x)
      return 45;
    if (-p4x<=x && x<p4y)
      return 46;
    if (p4y<=x && x<p3x)
      return 47;
    if (p3x<=x && x<p1y)
      return 48;
    if (p1y<=x && x<p4x)
      return 49;
    if (p4x<=x && x<p5x)
      return 50;
    if (p5x<=x && x<-p2x)
      return 51;
    if (-p2x<=x && x<-p1x)
      return 52;
    if (-p1x<=x && x<=p6x)
      return 53;
  }
  if (p3y<=y && y<p1y)
  {
    if (-p6x<=x && x<p1x)
      return 60;
    if (p1x<=x && x<p2x)
      return 61;
    if (p2x<=x && x<-p5x)
      return 62;
```

```
    if (-p5x<=x && x<-p4x)
      return 63;
    if (-p4x<=x && x<p4y)
      return 64;
    if (p4y<=x && x<p3x)
      return 65;
    if (p3x<=x && x<p1y)
      return 66;
    if (p1y<=x && x<p4x)
      return 67;
    if (p4x<=x && x<p5x)
      return 68;
    if (p5x<=x && x<-p2x)
      return 69;
    if (-p2x<=x && x<-p1x)
      return 70;
    if (-p1x<=x && x<=p6x)
      return 71;
  }
  if (p6x<x)
  {
    if (p4y<y && y<-p7y)
      return 54;
    if (-p7y<=y && y<p3y)
      return 56;
    if (p3y<=y && y<=p7y)
      return 58;
    if (p7y<y && y<p1y)
      return 72;
  }
  if (p1y<=y && y<-p2y)
  {
    if (-p6x>x)
      return 73;
    if (-p6x<=x && x<p1x)
      return 74;
    if (p1x<=x && x<p2x)
      return 75;
    if (p2x<=x && x<-p5x)
      return 76;
    if (-p5x<=x && x<-p4x)
      return 77;
    if (-p4x<=x && x<p4y)
      return 78;
    if (p4y<=x && x<p3x)
      return 79;
    if (p3x<=x && x<p1y)
      return 80;
```

```
    if (p1y<=x && x<p4x)
      return 81;
    if (p4x<=x && x<p5x)
      return 82;
    if (p5x<=x && x<-p2x)
      return 83;
    if (-p2x<=x && x<-p1x)
      return 84;
    if (-p1x<=x && x<=p6x)
      return 85;
    if (x>p6x)
      return 86;
  }
  if (-p2y<y && -p6x>x)
    return 87;
  if (-p2y<=y && y<=-p6y)
  {
    if (-p6x<=x && x<p1x)
      return 88;
    if (p1x<=x && x<p2x)
      return 89;
    if (p2x<=x && x<-p5x)
      return 90;
    if (-p5x<=x && x<-p4x)
      return 91;
    if (-p4x<=x && x<p4y)
      return 92;
    if (p4y<=x && x<p3x)
      return 93;
    if (p3x<=x && x<p1y)
      return 94;
    if (p1y<=x && x<p4x)
      return 95;
    if (p4x<=x && x<p5x)
      return 96;
    if (p5x<=x && x<-p2x)
      return 97;
    if (-p2x<=x && x<-p1x)
      return 98;
    if (-p1x<=x && x<=p6x)
      return 99;
  }
  if (-p2y<y && p6x<x)
    return 100;
  if (y>-p6y)
  {
    if (-p6x<=x && x<p1x)
      return 101;
```

```
      if (p1x<=x && x<p2x)
        return 102;
      if (p2x<=x && x<-p5x)
        return 103;
      if (-p5x<=x && x<-p4x)
        return 104;
      if (-p4x<=x && x<p4y)
        return 105;
      if (p4y<=x && x<p3x)
        return 106;
      if (p3x<=x && x<p1y)
        return 107;
      if (p1y<=x && x<p4x)
        return 108;
      if (p4x<=x && x<p5x)
        return 109;
      if (p5x<=x && x<-p2x)
        return 110;
      if (-p2x<=x && x<-p1x)
        return 111;
      if (-p1x<=x && x<=p6x)
        return 112;
   }
   return 0;
}
int Inf_Frame::retorna_quadrante_bola_fora_fundo()
{
   double x=coordenada_x_bola();
   double y=coordenada_y_bola();
   if (x>0.0)
   {
      if (y<=p2y)
        return 26;
      if (p2y<y && y<=p4y)
        return 40;
      if (p4y<y && y<=p3y)
        return 54;
      if (p3y<y && y<=p1y)
        return 72;
      if (p1y<y && y<=-p2y)
        return 86;
      if (-p2y<y)
        return 100;
   }
   else
   {
      if (y<=p2y)
        return 13;
```

```
      if (p2y<y && y<=p4y)
        return 27;
      if (p4y<y && y<=p3y)
        return 41;
      if (p3y<y && y<=p1y)
        return 59;
      if (p1y<y && y<=-p2y)
        return 73;
      if (-p2y<y)
        return 87;
   }
}
int Inf_Frame::retorna_quadrante_bola_fora_lateral()
{
   double x=coordenada_x_bola();
   double y=coordenada_y_bola();
   if (y<0.0)
   {
      if (-p6x<=x && x<p1x)
        return 1;
      if (p1x<=x && x<p2x)
        return 2;
      if (p2x<=x && x<-p5x)
        return 3;
      if (-p5x<=x && x<-p4x)
        return 4;
      if (-p4x<=x && x<p4y)
        return 5;
      if (p4y<=x && x<p3x)
        return 6;
      if (p3x<=x && x<p1y)
        return 7;
      if (p1y<=x && x<p4x)
        return 8;
      if (p4x<=x && x<p5x)
        return 9;
      if (p5x<=x && x<-p2x)
        return 10;
      if (-p2x<=x && x<-p1x)
        return 11;
      if (-p1x<=x && x<=p6x)
        return 12;
   }
   else
   {
      if (-p6x<=x && x<p1x)
        return 101;
      if (p1x<=x && x<p2x)
```

```
        return 102;
      if (p2x<=x && x<-p5x)
        return 103;
      if (-p5x<=x && x<-p4x)
        return 104;
      if (-p4x<=x && x<p4y)
        return 105;
      if (p4y<=x && x<p3x)
        return 106;
      if (p3x<=x && x<p1y)
        return 107;
      if (p1y<=x && x<p4x)
        return 108;
      if (p4x<=x && x<p5x)
        return 109;
      if (p5x<=x && x<-p2x)
        return 110;
      if (-p2x<=x && x<-p1x)
        return 111;
      if (-p1x<=x && x<=p6x)
        return 112;
   }
}
Ponto2D Inf_Frame::consultar_coordenadas_bola ()
{
   return coordenadas_bola;
}
Ponto2D Inf_Frame::consultar_coordenadas_jogador (bool oponente , int jogador)
{
   if (oponente)
      return coordenadas_time_oponente[jogador-1];
   else
      return coordenadas_time[jogador-1];
}
void Inf_Frame::setar_coordenadas_jogador (bool oponente , int jogador, double x,
double y)
{
   if (oponente)
      coordenadas_time_oponente[jogador-1].setar_xy(x, y);
   else
      coordenadas_time[jogador-1].setar_xy(x, y);
}
void Inf_Frame::setar_coordenadas_bola (double x, double y)
{
      coordenadas_bola.setar_xy(x, y);
}
void Inf_Frame::setar_bola_com_oponente(bool posse)
{
```

```
  bola_com_oponente=posse;
}
void Inf_Frame::setar_jogador_com_posse_da_bola(int jogador)
{
  jogador_com_posse_da_bola=jogador;
}
double Inf_Frame::coordenada_x_bola ()
{
  return (coordenadas_bola.consultar_x());
}
double Inf_Frame::coordenada_y_bola ()
{
  return (coordenadas_bola.consultar_y());
}
int Inf_Frame::consultar_jogador_com_posse_da_bola ()
{
  return (jogador_com_posse_da_bola);
}
bool Inf_Frame::posse_bola_oponente ()
{
  return (bola_com_oponente);
}
int Inf_Frame::jogador_mais_proximo_da_bola (bool oponente)
{
  double d = consultar_coordenadas_jogador(oponente,1).distancia_ao_ponto
(consultar_coordenadas_bola()), d_aux;
  int jogador=1;
  for (int i=2; i<12; i++)
  {
    d_aux = consultar_coordenadas_jogador(oponente,i).distancia_ao_ponto
(consultar_coordenadas_bola());
    if (d_aux<d)
    {
      d=d_aux;
      jogador=i;
    }
  }
  return jogador;
}
class Historico
{
  char nome_time[40];
  bool oponente_no_campo_esquerdo;
  int ciclo;
  Inf_Frame frame1;
  Inf_Frame frame2;
  Inf_Frame frame3;
  Inf_Frame frame4;
```

```
  Inf_Frame instante_do_ultimo_contato_do_jogador_com_a_bola;
  Inf_Frame instante_que_o_jogador_adquiriu_a_posse_de_bola;
  Inf_Frame antigo_instante_do_ultimo_contato_do_jogador_com_a_bola;
  Inf_Frame antigo_instante_que_o_jogador_adquiriu_a_posse_de_bola;
  Inf_Frame bkp_frame1;
  Inf_Frame bkp_frame2;
  Inf_Frame bkp_instante_do_ultimo_contato_do_jogador_com_a_bola;
  Inf_Frame bkp_instante_que_o_jogador_adquiriu_a_posse_de_bola;
  Estado estado;
  bool bola_mantem_direcao_e_sentido();
  bool oponente_esta_no_campo_esquerdo ();
 public:
  void inicializar_historico(char *nome);
  int atualizar_historico (char *visao_global);
  Inf_Frame retorna_frame(int n);
  void restaurar_historico();
  bool atualizar_informacoes_para_analise_da_ocorrencia_de_conducao();
  int consultar_ciclo();
};
void Historico::inicializar_historico(char *nome)
{
  for(int i=0; nome_time[i]=nome[i]; i++);
  estado=pre_inicial;
}
bool Historico::atualizar_informacoes_para_analise_da_ocorrencia_de_conducao()
{
  bool retorno;
  if (retorno=(instante_que_o_jogador_adquiriu_a_posse_de_bola.consultar_
jogador_com_posse_da_bola()==bkp_instante_que_o_jogador_adquiriu_a_posse_d
e_bola.consultar_jogador_com_posse_da_bola()))
  {
    if (instante_do_ultimo_contato_do_jogador_com_a_bola.consultar_
coordenadas_jogador (true, instante_do_ultimo_contato_do_jogador_com_a_
bola.consultar_jogador_com_posse_da_bola()).distancia_ao_ponto(instante_do_
ultimo_contato_do_jogador_com_a_bola.consultar_coordenadas_bola())>=bkp_insta
nte_do_ultimo_contato_do_jogador_com_a_bola.consultar_coordenadas_jogador(tru
e,bkp_instante_do_ultimo_contato_do_jogador_com_a_bola.consultar_jogador_com
_posse_da_bola()).distancia_ao_ponto(bkp_instante_do_ultimo_contato_do_jogador
_com_a_bola.consultar_coordenadas_bola()))
      instante_do_ultimo_contato_do_jogador_com_a_bola = bkp_instante_do_
ultimo_contato_do_jogador_com_a_bola;
    instante_que_o_jogador_adquiriu_a_posse_de_bola = bkp_instante_que_o_
jogador_adquiriu_a_posse_de_bola;
  }
  return retorno;
}
void Historico::restaurar_historico()
{
```

```
  frame1.setar_jogador_com_posse_da_bola(bkp_frame1.consultar_jogador_
com_posse_da_bola());
  frame1.setar_bola_com_oponente(bkp_frame1.posse_bola_oponente());
  frame2.setar_jogador_com_posse_da_bola(bkp_frame2.consultar_jogador_
com_posse_da_bola());
  frame2.setar_bola_com_oponente(bkp_frame2.posse_bola_oponente());
  instante_do_ultimo_contato_do_jogador_com_a_bola=bkp_instante_do_ultimo_
contato_do_jogador_com_a_bola;
  instante_que_o_jogador_adquiriu_a_posse_de_bola=bkp_instante_que_o_
jogador_adquiriu_a_posse_de_bola;
}
int Historico::consultar_ciclo()
{
  return ciclo;
}
Inf_Frame Historico::retorna_frame(int n)
{
  switch (n)
  {
    case 1:
      return frame1;
    case 2:
      return frame2;
    case 3:
      return instante_que_o_jogador_adquiriu_a_posse_de_bola;
    case 4:
      return instante_do_ultimo_contato_do_jogador_com_a_bola;
    case 5:
      return antigo_instante_que_o_jogador_adquiriu_a_posse_de_bola;
    case 6:
      return antigo_instante_do_ultimo_contato_do_jogador_com_a_bola;
  }
}
bool Historico::oponente_esta_no_campo_esquerdo ()
{
  return (oponente_no_campo_esquerdo);
}
bool Historico::bola_mantem_direcao_e_sentido()
{
  double p1_x=frame1.coordenada_x_bola();
  double p1_y=frame1.coordenada_y_bola();
  double p2_x=frame2.coordenada_x_bola();
  double p2_y=frame2.coordenada_y_bola();
  double p3_x=frame4.coordenada_x_bola();
  double p3_y=frame4.coordenada_y_bola();
  if (p1_x==p2_x && p1_y==p2_y)
    return (true);
  if (p1_x==p3_x && p2_x==p3_x && p1_y==p3_y && p2_y==p3_y)
```

```
      return (true);
   if ((p1_x==p3_x && p2_x==p3_x) && (p1_y<p2_y?p3_y>=p2_y:p3_y<=p2_y))
      return (true);
   if ((p1_y==p3_y && p2_y==p3_y) && (p1_x<p2_x?p3_x>=p2_x:p3_x<=p2_x))
      return (true);
   if (pow(p3_x-p2_x,2)>=pow(p3_y-p2_y,2))
   {
      if ((p3_y-p1_y)/(p3_x-p1_x)-(p3_y-p2_y)/(p3_x-p2_x)<tolerancia)
         if (-tolerancia<(p3_y-p1_y)/(p3_x-p1_x)-(p3_y-p2_y)/(p3_x-p2_x))
            if (p1_x<p2_x?p3_x>=p2_x:p3_x<=p2_x)
               return (true);
   }
   else
   {
      if ((p3_y-p1_y)/(p3_x-p1_x)-(p3_y-p2_y)/(p3_x-p2_x)<tolerancia)
         if (-tolerancia<(p3_y-p1_y)/(p3_x-p1_x)-(p3_y-p2_y)/(p3_x-p2_x))
            if (p1_y<p2_y?p3_y>=p2_y:p3_y<=p2_y)
               return (true);
   }
   return (false);
}
int Historico::atualizar_historico (char *visao_global)
{
   double lixo;
   char ntime[40];
   char str_lixo[40];
   double aux[44];
   double coordenadas_bola[2];
   int ciclo_corrente;
   int retorno=0;
   sscanf (visao_global, "(see_global %d ((g l) %lf %lf) ((g r) %lf %lf) ((b) %lf %lf %lf
%lf) ((p %s 1 goalie) %lf %lf %lf %lf %lf %lf) ((p %s 2) %lf %lf %lf %lf %lf %lf) ((p %s
3) %lf %lf %lf %lf %lf %lf) ((p %s 4) %lf %lf %lf %lf %lf %lf) ((p %s 5) %lf %lf %lf %lf
%lf %lf) ((p %s 6) %lf %lf %lf %lf %lf %lf) ((p %s 7) %lf %lf %lf %lf %lf %lf) ((p %s 8)
%lf %lf %lf %lf %lf %lf) ((p %s 9) %lf %lf %lf %lf %lf %lf) ((p %s 10) %lf %lf %lf %lf
%lf %lf) ((p %s 11) %lf %lf %lf %lf %lf %lf) ((p %s 1 goalie) %lf %lf %lf %lf %lf %lf) ((p
%s 2) %lf %lf %lf %lf %lf %lf) ((p %s 3) %lf %lf %lf %lf %lf %lf) ((p %s 4) %lf %lf %lf
%lf %lf %lf) ((p %s 5) %lf %lf %lf %lf %lf %lf) ((p %s 6) %lf %lf %lf %lf %lf %lf) ((p %s
7) %lf %lf %lf %lf %lf %lf) ((p %s 8) %lf %lf %lf %lf %lf %lf) ((p %s 9) %lf %lf %lf %lf
%lf %lf) ((p %s 10) %lf %lf %lf %lf %lf %lf) ((p %s 11) %lf %lf %lf %lf %lf %lf))",
&ciclo_corrente,&lixo,&lixo,&lixo,&lixo,&coordenadas_bola[0],&coordenadas_bola[1],
&lixo,&lixo,ntime,aux,aux+1,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+2,aux+3,&lixo,&lixo,&l
ixo,&lixo,str_lixo,aux+4,aux+5,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+6,aux+7,&lixo,&lixo,
&lixo,&lixo,str_lixo,aux+8,aux+9,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+10,aux+11,&lixo,
&lixo,&lixo,&lixo,str_lixo,aux+12,aux+13,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+14,aux+1
5,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+16,aux+17,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+18
,aux+19,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+20,aux+21,&lixo,&lixo,&lixo,&lixo,str_lixo,
aux+22,aux+23,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+24,aux+25,&lixo,&lixo,&lixo,&lixo,
```

```
str_lixo,aux+26,aux+27,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+28,aux+29,&lixo,&lixo,&lix
o,&lixo,str_lixo,aux+30,aux+31,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+32,aux+33,&lixo,&l
ixo,&lixo,&lixo,str_lixo,aux+34,aux+35,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+36,aux+37,
&lixo,&lixo,&lixo,&lixo,str_lixo,aux+38,aux+39,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+40,a
ux+41,&lixo,&lixo,&lixo,&lixo,str_lixo,aux+42,aux+43,&lixo,&lixo,&lixo,&lixo);
  if (ciclo_corrente>=3000 && ciclo<3000)
  {
    for (int i=0; i<11; i++)
    {
      frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
      frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
    }
    frame1.setar_coordenadas_bola(0, 0);
    estado=inicio_2_tempo;
    ciclo=ciclo_corrente;
    return 5;
  }
  switch (estado)
  {
    case pre_inicial:
    {
      if (!strcmp(ntime,nome_time))
        oponente_no_campo_esquerdo=false;
      else
        oponente_no_campo_esquerdo=true;
      ciclo=ciclo_corrente;
      estado=inicial;
      retorno=-1;
      break;
    }
    case inicial:
    {
      if (ciclo_corrente)
      {
        for (int i=0; i<11; i++)
        {
          frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
          frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
        }
        frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
        if (frame1.consultar_coordenadas_jogador(true,frame1.jogador_mais_
proximo_da_bola(true)).distancia_ao_ponto(frame1.consultar_coordenadas_bola())<
```

```
frame1.consultar_coordenadas_jogador(false,frame1.jogador_mais_proximo_da_bol
a(false)).distancia_ao_ponto(frame1.consultar_coordenadas_bola()))
            frame1.setar_bola_com_oponente(true);
          else
            frame1.setar_bola_com_oponente(false);
          frame1.setar_jogador_com_posse_da_bola
(frame1.jogador_mais_proximo_da_bola(frame1.posse_bola_oponente()));
          ciclo=ciclo_corrente;
          estado=pos_inicial;
          if (frame1.posse_bola_oponente())
          {
            instante_do_ultimo_contato_do_jogador_com_a_bola=frame1;
            instante_que_o_jogador_adquiriu_a_posse_de_bola=frame1;
            retorno=1;
          }
          else
            retorno=4;
        }
        break;
      }
      case pos_gol:
      {
        Circulo c(5.0);
        Ponto2D p(coordenadas_bola[0],coordenadas_bola[1]);
        if (ciclo<ciclo_corrente && c.ponto_pertence_ao_circulo(p))
        {
          for (int i=0; i<11; i++)
          {
            frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
            frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
          }
          frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
          if          (frame1.consultar_coordenadas_jogador(true,frame1.jogador_mais_
proximo_da_bola(true)).distancia_ao_ponto(frame1.consultar_coordenadas_bola())<
frame1.consultar_coordenadas_jogador(false,frame1.jogador_mais_proximo_da_bol
a(false)).distancia_ao_ponto(frame1.consultar_coordenadas_bola()))
            frame1.setar_bola_com_oponente(true);
          else
            frame1.setar_bola_com_oponente(false);
          frame1.setar_jogador_com_posse_da_bola(frame1.jogador_mais_proximo_
da_bola(frame1.posse_bola_oponente()));
          ciclo=ciclo_corrente;
          estado=pos_inicial;
          if (frame1.posse_bola_oponente())
          {
```

```
            instante_do_ultimo_contato_do_jogador_com_a_bola=frame1;
            instante_que_o_jogador_adquiriu_a_posse_de_bola=frame1;
            retorno=1;
          }
          else
            retorno=4;
          estado=pos_inicial;
        }
        else
        {
          ciclo=ciclo_corrente;
          retorno=13;
        }
        break;
      }
      case inicio_2_tempo:
      {
        frame1.setar_bola_com_oponente(!oponente_no_campo_esquerdo);
        frame1.setar_jogador_com_posse_da_bola(frame1.jogador_mais_
proximo_da_bola(!oponente_no_campo_esquerdo));
        if (frame1.posse_bola_oponente())
        {
          instante_do_ultimo_contato_do_jogador_com_a_bola=frame1;
          instante_que_o_jogador_adquiriu_a_posse_de_bola=frame1;
          retorno=1;
        }
        else
          retorno=4;
        for (int i=0; i<11; i++)
        {
          frame2.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
          frame2.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
        }
        frame2.setar_jogador_com_posse_da_bola(frame1.consultar_jogador_com_
posse_da_bola());
        frame2.setar_bola_com_oponente(frame1.posse_bola_oponente());
        frame2.setar_coordenadas_bola(coordenadas_bola[0], coordenadas_bola[1]);
        ciclo=ciclo_corrente;
        estado=corrente;
        break;
      }
      case pos_saida_de_bola_fundo:
      {
        if ((coordenadas_bola[0]!=47 && coordenadas_bola[0]!=-47) ||
        (coordenadas_bola[1]!=9.16 && coordenadas_bola[1]!=-9.16))
        {
```

```
          if          (frame1.consultar_coordenadas_jogador(true,frame1.jogador_mais_
proximo_da_bola(true)).distancia_ao_ponto(frame1.consultar_coordenadas_bola())<
frame1.consultar_coordenadas_jogador(false,frame1.jogador_mais_proximo_da_bol
a(false)).distancia_ao_ponto(frame1.consultar_coordenadas_bola()))
            frame1.setar_bola_com_oponente(true);
          else
            frame1.setar_bola_com_oponente(false);
          frame1.setar_jogador_com_posse_da_bola(frame1.jogador_mais_
proximo_da_bola(frame1.posse_bola_oponente()));
          if (frame1.posse_bola_oponente())
          {
            instante_do_ultimo_contato_do_jogador_com_a_bola=frame1;
            instante_que_o_jogador_adquiriu_a_posse_de_bola=frame1;
            retorno=1;
          }
          else
            retorno=4;
          for (int i=0; i<11; i++)
          {
            frame2.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
            frame2.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
          }
          frame2.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
          frame2.setar_bola_com_oponente(frame1.posse_bola_oponente());
          frame2.setar_jogador_com_posse_da_bola(frame1.consultar_jogador_com_
posse_da_bola());
          ciclo=ciclo_corrente;
          estado=corrente;
        }
        else
        {
          for (int i=0; i<11; i++)
          {
            frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
            frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
          }
          frame1.setar_coordenadas_bola(coordenadas_bola[0],coordenadas_bola[1]);
          ciclo=ciclo_corrente;
          retorno=13;
        }
        break;
      }
      case pos_saida_de_bola_lateral:
```

```
    {
      if (coordenadas_bola[1]!=p6y && coordenadas_bola[1]!=-p6y)
      {
        if (frame1.consultar_coordenadas_jogador(true,frame1.jogador_mais_
proximo_da_bola(true)).distancia_ao_ponto(frame1.consultar_coordenadas_bola())<
frame1.consultar_coordenadas_jogador(false,frame1.jogador_mais_proximo_da_bol
a(false)).distancia_ao_ponto(frame1.consultar_coordenadas_bola()))
          frame1.setar_bola_com_oponente(true);
        else
          frame1.setar_bola_com_oponente(false);
        frame1.setar_jogador_com_posse_da_bola(frame1.jogador_mais_proximo_
da_bola(frame1.posse_bola_oponente()));
        if (frame1.posse_bola_oponente())
        {
          instante_do_ultimo_contato_do_jogador_com_a_bola=frame1;
          instante_que_o_jogador_adquiriu_a_posse_de_bola=frame1;
          retorno=1;
        }
        else
          retorno=4;
        for (int i=0; i<11; i++)
        {
          frame2.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
          frame2.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
        }
        frame2.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
        frame2.setar_bola_com_oponente(frame1.posse_bola_oponente());
        frame2.setar_jogador_com_posse_da_bola(frame1.consultar_jogador_
com_posse_da_bola());
        ciclo=ciclo_corrente;
        estado=corrente;
      }
      else
      {
        for (int i=0; i<11; i++)
        {
          frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
          frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
        }
        frame1.setar_coordenadas_bola(coordenadas_bola[0],coordenadas_bola[1]);
        ciclo=ciclo_corrente;
        retorno=13;
      }
```

```
      break;
    }
    case pos_inicial:
    {
      if (ciclo<ciclo_corrente)
      {
        for (int i=0; i<11; i++)
        {
          frame2.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
          frame2.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
        }
        frame2.setar_jogador_com_posse_da_bola(frame1.consultar_jogador_com_
posse_da_bola());
        frame2.setar_bola_com_oponente(frame1.posse_bola_oponente());
        frame2.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
        ciclo=ciclo_corrente;
        estado=corrente;
        if (frame1.posse_bola_oponente())
          retorno=7;
        else
          retorno=6;
      }
      break;
    }
    case corrente:
    {
      if (ciclo<ciclo_corrente)
      {
        if (((coordenadas_bola[0]<-p6x && frame2.coordenada_x_bola()>=-p6x) ||
(coordenadas_bola[0]>p6x && frame2.coordenada_x_bola()<=p6x)) &&
coordenadas_bola[1]<10 && coordenadas_bola[1]>-10)
        {
          for (int i=0; i<11; i++)
          {
            frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
            frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
          }
          frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
          ciclo=ciclo_corrente;
          estado=pos_gol;
          if ((coordenadas_bola[0]<-p6x && !oponente_no_campo_esquerdo) ||
(coordenadas_bola[0]>p6x && oponente_no_campo_esquerdo))
```

```
            retorno=8;
          else
            retorno=9;
        }
        else
          if ((coordenadas_bola[1]==p6y && frame2.coordenada_y_bola()==p6y) ||
(coordenadas_bola[1]==-p6y && frame2.coordenada_y_bola()==-p6y))
          {
            frame2=frame1;
            for (int i=0; i<11; i++)
            {
              frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,
i+1,*(aux+i*2),*(aux+i*2+1));
              frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,
i+1,*(aux+i*2+22),*(aux+i*2+23));
            }
            frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
            ciclo=ciclo_corrente;
            estado=pos_saida_de_bola_lateral;
            retorno=11;
          }
          else
            if ((((coordenadas_bola[0]==47 && coordenadas_bola[1]==9.16) ||
(coordenadas_bola[0]==47 && coordenadas_bola[1]==-9.16))&&
frame2.coordenada_x_bola()>50) ||
            (((coordenadas_bola[0]==-47 && coordenadas_bola[1]==9.16) ||
(coordenadas_bola[0]==-47 && coordenadas_bola[1]==-9.16))&&
frame2.coordenada_x_bola()<-50))
            {
              frame2=frame1;
              for (int i=0; i<11; i++)
              {
                frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,
i+1,*(aux+i*2),*(aux+i*2+1));
                frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,
i+1,*(aux+i*2+22),*(aux+i*2+23));
              }
              frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
              ciclo=ciclo_corrente;
              estado=pos_saida_de_bola_fundo;
              retorno=12;
            }
            else
            {
              for (int i=0; i<11; i++)
              {
```

```
                frame4.setar_coordenadas_jogador(oponente_no_campo_esquerdo,
i+1,*(aux+i*2),*(aux+i*2+1));
                frame4.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,
i+1,*(aux+i*2+22),*(aux+i*2+23));
              }
              frame4.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
              ciclo=ciclo_corrente;
              if (bola_mantem_direcao_e_sentido())
              {
                frame4.setar_jogador_com_posse_da_bola(frame2.consultar_
jogador_com_posse_da_bola());
                frame4.setar_bola_com_oponente(frame2.posse_bola_oponente());
                frame1=frame2;
                frame2=frame4;
                if (frame1.posse_bola_oponente())
                {
                  retorno=7;
                  if  (instante_do_ultimo_contato_do_jogador_com_a_bola.consultar_
coordenadas_jogador(true,instante_do_ultimo_contato_do_jogador_com_a_bola.con
sultar_jogador_com_posse_da_bola()).distancia_ao_ponto(instante_do_ultimo_cont
ato_do_jogador_com_a_bola.consultar_coordenadas_bola())>=
frame4.consultar_coordenadas_jogador(true,frame4.consultar_jogador_com_posse_
da_bola()).distancia_ao_ponto(frame4.consultar_coordenadas_bola()))
                    instante_do_ultimo_contato_do_jogador_com_a_bola=frame4;
                }
                else
                  retorno=6;
              }
              else
              {
                frame3=frame4;
                estado=analise;
                retorno=0;
              }
            }
      }
      break;
    }
    case analise:
    {
      if (ciclo<ciclo_corrente)
      {
        if  (((coordenadas_bola[0]<-p6x  &&  frame3.coordenada_x_bola()>=-p6x)  ||
(coordenadas_bola[0]>p6x  &&  frame3.coordenada_x_bola()<=p6x))  &&
coordenadas_bola[1]<10 && coordenadas_bola[1]>-10)
        {
          for (int i=0; i<11; i++)
```

```
            {
              frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,i+1,
*(aux+i*2),*(aux+i*2+1));
              frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,i+1,
*(aux+i*2+22),*(aux+i*2+23));
            }
            frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
            ciclo=ciclo_corrente;
            estado=pos_gol;
            if ((coordenadas_bola[0]<-p6x && !oponente_no_campo_esquerdo) ||
(coordenadas_bola[0]>p6x && oponente_no_campo_esquerdo))
              retorno=8;
            else
              retorno=9;
          }
          else
            if ((coordenadas_bola[1]==p6y && frame3.coordenada_y_bola()==p6y) ||
(coordenadas_bola[1]==-p6y && frame3.coordenada_y_bola()==-p6y))
            {
              for (int i=0; i<11; i++)
              {
                frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,
i+1,*(aux+i*2),*(aux+i*2+1));
                frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,
i+1,*(aux+i*2+22),*(aux+i*2+23));
              }
              frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
              ciclo=ciclo_corrente;
              estado=pos_saida_de_bola_lateral;
              retorno=11;
            }
            else
              if ((((coordenadas_bola[0]==47 && coordenadas_bola[1]==9.16) ||
(coordenadas_bola[0]==47 && coordenadas_bola[1]==-9.16))&&
frame3.coordenada_x_bola()>50) ||
              (((coordenadas_bola[0]==-47 && coordenadas_bola[1]==9.16) ||
(coordenadas_bola[0]==-47 && coordenadas_bola[1]==-9.16))&&
frame3.coordenada_x_bola()<-50))
              {
                for (int i=0; i<11; i++)
                {
                  frame1.setar_coordenadas_jogador(oponente_no_campo_esquerdo,
i+1,*(aux+i*2),*(aux+i*2+1));
                  frame1.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,
i+1,*(aux+i*2+22),*(aux+i*2+23));
                }
```

```
                frame1.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
                ciclo=ciclo_corrente;
                estado=pos_saida_de_bola_fundo;
                retorno=12;
              }
              else
              {
                for (int i=0; i<11; i++)
                {
                  frame4.setar_coordenadas_jogador(oponente_no_campo_esquerdo,
i+1,*(aux+i*2),*(aux+i*2+1));
                  frame4.setar_coordenadas_jogador(!oponente_no_campo_esquerdo,
i+1,*(aux+i*2+22),*(aux+i*2+23));
                }
                frame4.setar_coordenadas_bola(coordenadas_bola[0],
coordenadas_bola[1]);
                ciclo=ciclo_corrente;
                Reta r1(frame1.consultar_coordenadas_bola(),frame2.consultar_
coordenadas_bola()),r2(frame3.consultar_coordenadas_bola(),frame4.consultar_coo
rdenadas_bola());
                Ponto2D interseccao_das_trajetorias=r1.ponto_de_interseccao_com_
a_reta(r2);
                Circulo area_de_analise (interseccao_das_trajetorias,
frame2.consultar_coordenadas_bola().distancia_ao_ponto(interseccao_das_trajetori
as)+2);
                int jogador_oponente_mais_proximo_da_interseccao_das_trajetorias=
0;
                double distancia_entre_jogador_oponente_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias;
                for (int i=1; i<12; i++)
                {
                  if (area_de_analise.ponto_pertence_ao_circulo
(frame2.consultar_coordenadas_jogador(true,i)) ||
area_de_analise.ponto_pertence_ao_circulo(frame3.consultar_coordenadas_jogador
(true,i)))
                  {
                    Reta r (frame2.consultar_coordenadas_jogador(true,i),
frame3.consultar_coordenadas_jogador(true,i));
                    double distancia=r.distancia_ao_ponto(interseccao_das_trajetorias);
                    if
(!jogador_oponente_mais_proximo_da_interseccao_das_trajetorias)
                    {
                      distancia_entre_jogador_oponente_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias=distancia;
                      jogador_oponente_mais_proximo_da_interseccao_das_
trajetorias=i;
                    }
```

```
                    else
                      if    (distancia<distancia_entre_jogador_oponente_mais_proximo_
da_interseccao_das_trajetorias_e_a_interseccao_das_trajetorias)
                      {
                        distancia_entre_jogador_oponente_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias=distancia;
                        jogador_oponente_mais_proximo_da_interseccao_das_
trajetorias=i;
                      }
                    }
                  }
                  int jogador_mais_proximo_da_interseccao_das_trajetorias=0;
                  double    distancia_entre_jogador_mais_proximo_da_interseccao_das_
trajetorias_e_a_interseccao_das_trajetorias;
                  for (int i=1; i<12; i++)
                  {
                    if      (area_de_analise.ponto_pertence_ao_circulo(frame2.consultar_
coordenadas_jogador(false,i))                                                    ||
area_de_analise.ponto_pertence_ao_circulo(frame3.consultar_coordenadas_jogador
(false,i)))
                    {
                      Reta                  r(frame2.consultar_coordenadas_jogador(false,i),
frame3.consultar_coordenadas_jogador(false,i));
                      double distancia=r.distancia_ao_ponto(interseccao_das_trajetorias);
                      if (!jogador_mais_proximo_da_interseccao_das_trajetorias)
                      {
                        distancia_entre_jogador_mais_proximo_da_interseccao_
das_trajetorias_e_a_interseccao_das_trajetorias=distancia;
                        jogador_mais_proximo_da_interseccao_das_trajetorias=i;
                      }
                      else
                        if            (distancia<distancia_entre_jogador_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias)
                        {
                          distancia_entre_jogador_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias=distancia;
                          jogador_mais_proximo_da_interseccao_das_trajetorias=i;
                        }
                    }
                  }
                  if      (jogador_oponente_mais_proximo_da_interseccao_das_trajetorias
&& jogador_mais_proximo_da_interseccao_das_trajetorias)
                  {
                    if (!frame2.posse_bola_oponente())
                    {
                      if            (distancia_entre_jogador_oponente_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias<distancia_entre_jogad
or_mais_proximo_da_interseccao_das_trajetorias_e_a_interseccao_das_trajetorias)
```

```
                    {
                      frame3. setar_bola_com_oponente(true);
                      frame4. setar_bola_com_oponente(true);
                      frame3.setar_jogador_com_posse_da_bola(jogador_
oponente_mais_proximo_da_interseccao_das_trajetorias);
                      frame4.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                      retorno=10;
                      instante_do_ultimo_contato_do_jogador_com_a_bola=frame3;
                      instante_que_o_jogador_adquiriu_a_posse_de_bola=frame3;
                    }
                    else
                    {
                      frame3. setar_bola_com_oponente(false);
                      frame4. setar_bola_com_oponente(false);
                      frame3.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                      frame4.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                      retorno=6;
                    }
                  }
                  else
                  {
                    if (distancia_entre_jogador_oponente_mais_proximo_da_
interseccao_das_trajetorias_e_a_interseccao_das_trajetorias<distancia_entre_jogad
or_mais_proximo_da_interseccao_das_trajetorias_e_a_interseccao_das_trajetorias)
                    {
                      if (jogador_oponente_mais_proximo_da_interseccao_das_
trajetorias!=frame2.consultar_jogador_com_posse_da_bola())
                      {
                        frame3.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                        frame4.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                        retorno=3;
                        antigo_instante_do_ultimo_contato_do_jogador_com_a_bola=
instante_do_ultimo_contato_do_jogador_com_a_bola;
                        antigo_instante_que_o_jogador_adquiriu_a_posse_de_bola=
instante_que_o_jogador_adquiriu_a_posse_de_bola;
                        instante_do_ultimo_contato_do_jogador_com_a_bola=frame3;
                        instante_que_o_jogador_adquiriu_a_posse_de_bola=frame3;
                      }
                      else
                      {
                        frame3. setar_bola_com_oponente(true);
                        frame4. setar_bola_com_oponente(true);
```

```
                    frame3.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                    frame4.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                    retorno=7;
                    instante_do_ultimo_contato_do_jogador_com_a_bola=frame4;
                  }
                }
                else
                {
                  frame3.setar_jogador_com_posse_da_bola(jogador_mais_
proximo_da_interseccao_das_trajetorias);
                  frame4.setar_jogador_com_posse_da_bola(jogador_mais_
proximo_da_interseccao_das_trajetorias);
                  frame3.setar_bola_com_oponente(false);
                  frame4.setar_bola_com_oponente(false);
                  retorno=2;
                  bkp_frame1=frame1;
                  bkp_frame2=frame2;
                  bkp_instante_do_ultimo_contato_do_jogador_com_a_bola=
instante_do_ultimo_contato_do_jogador_com_a_bola;
                  bkp_instante_que_o_jogador_adquiriu_a_posse_de_bola=
instante_que_o_jogador_adquiriu_a_posse_de_bola;
                  }
                }
              }
              else
              {
                if  (jogador_oponente_mais_proximo_da_interseccao_das_trajetorias
&& !jogador_mais_proximo_da_interseccao_das_trajetorias)
                {
                  if (!frame2.posse_bola_oponente())
                  {
                    frame3. setar_bola_com_oponente(true);
                    frame4. setar_bola_com_oponente(true);
                    frame3.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                    frame4.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                    retorno=10;
                    instante_do_ultimo_contato_do_jogador_com_a_bola=frame3;
                    instante_que_o_jogador_adquiriu_a_posse_de_bola=frame3;
                  }
                  else
                  {
                    if          (jogador_oponente_mais_proximo_da_interseccao_das_
trajetorias!=frame2.consultar_jogador_com_posse_da_bola())
                    {
```

```
                    frame3.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                    frame4.setar_jogador_com_posse_da_bola(jogador_oponente_
mais_proximo_da_interseccao_das_trajetorias);
                    retorno=3;
                    antigo_instante_do_ultimo_contato_do_jogador_com_a_bola=
instante_do_ultimo_contato_do_jogador_com_a_bola;
                    antigo_instante_que_o_jogador_adquiriu_a_posse_de_bola=
instante_que_o_jogador_adquiriu_a_posse_de_bola;
                    instante_do_ultimo_contato_do_jogador_com_a_bola=frame3;
                    instante_que_o_jogador_adquiriu_a_posse_de_bola=frame3;
                  }
                  else
                  {
                    frame3. setar_bola_com_oponente(true);
                    frame4. setar_bola_com_oponente(true);
                    frame3.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                    frame4.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                    retorno=7;
                    instante_do_ultimo_contato_do_jogador_com_a_bola=frame4;
                  }
                }
              }
              else
              {
                if          (!jogador_oponente_mais_proximo_da_interseccao_das_
trajetorias && jogador_mais_proximo_da_interseccao_das_trajetorias)
                {
                  if (!frame2.posse_bola_oponente())
                  {
                    frame3. setar_bola_com_oponente(false);
                    frame4. setar_bola_com_oponente(false);
                    frame3.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                    frame4.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                    retorno=6;
                  }
                  else
                  {
                    frame3.setar_jogador_com_posse_da_bola(jogador_mais_
proximo_da_interseccao_das_trajetorias);
                    frame4.setar_jogador_com_posse_da_bola(jogador_mais_
proximo_da_interseccao_das_trajetorias);
                    frame3.setar_bola_com_oponente(false);
                    frame4.setar_bola_com_oponente(false);
```

```
                        retorno=2;
                        bkp_frame1=frame1;
                        bkp_frame2=frame2;
                        bkp_instante_do_ultimo_contato_do_jogador_com_a_bola=
instante_do_ultimo_contato_do_jogador_com_a_bola;
                        bkp_instante_que_o_jogador_adquiriu_a_posse_de_bola=
instante_que_o_jogador_adquiriu_a_posse_de_bola;
                      }
                    }
                    else
                    {
                      if (frame2.posse_bola_oponente())
                      {
                        frame3. setar_bola_com_oponente(true);
                        frame4. setar_bola_com_oponente(true);
                        frame3.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                        frame4.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                        retorno=7;
                        instante_do_ultimo_contato_do_jogador_com_a_bola=frame4;
                      }
                      else
                      {
                        frame3. setar_bola_com_oponente(false);
                        frame4. setar_bola_com_oponente(false);
                        frame3.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                        frame4.setar_jogador_com_posse_da_bola(frame1.consultar_
jogador_com_posse_da_bola());
                        retorno=6;
                      }
                    }
                  }
                }
                frame1=frame3;
                frame2=frame4;
                estado=corrente;
              }
        }
        break;
      }
    }
    return retorno;
}
class RdP
{
    struct elemento
```

```
  {
    int i;
    int j;
    int valor;
    int ocorrencia;
    struct elemento *PC;
    struct elemento *PL;
        };
  struct lugar
  {
    char descricao[7];
    elemento *PC;
    struct lugar *prox;
  };
  struct transicao
  {
    Transicao descricao;
    char lugar_origem[7];
    char lugar_destino[7];
    elemento *PL;
    struct transicao *prox;
  };
  struct matriz
  {
    struct lugar *lugares;
    struct transicao *transicoes;
  };
  struct matriz ME;
  struct matriz MS;
  int localizar_posicao_de_insercao_lugar(char desc[7]);
  int localizar_posicao_de_insercao_transicao(Transicao desc, char lo[7], char ld[7]);
  int localizar_indice_do_lugar(char desc[7]);
  int localizar_indice_da_transicao(Transicao desc, char lo[7], char ld[7]);
public:
  RdP();
  ~RdP();
  void inserir_lugar(Inf_Frame frame, Situacao s);
  void inserir_transicao(Transicao desc, Inf_Frame frame_LO, Inf_Frame frame_LD,
Situacao s);
  void escrever_RdP_em_arquivo (FILE *pf);
  void escrever_RdP_com_frequencia_em_arquivo (FILE *pf);
};
RdP::RdP()
{
  ME.lugares = NULL;
  ME.transicoes = NULL;
  MS.lugares = NULL;
  MS.transicoes = NULL;
```

```
}
RdP::~RdP()
{
  struct lugar *r, *t=ME.lugares;
  elemento *p, *q ;
  while (t!=NULL)
  {
    p=(*t).PC;
    while(p!=NULL)
    {
      q = p;
      p= p->PC;
      delete(q);
    }
    r = t;
    t = t->prox;
    delete(r);
  }
  struct transicao *u, *v=ME.transicoes;
  while(v!=NULL)
  {
    u = v;
    v= v->prox;
    delete(u);
  }
  t=MS.lugares;
  while (t!=NULL)
  {
    p=(*t).PC;
    while(p!=NULL)
    {
      q = p;
      p= p->PC;
      delete(q);
    }
    r = t;
    t = t->prox;
    delete(r);
  }
  v=MS.transicoes;
  while(v!=NULL)
  {
    u = v;
    v= v->prox;
    delete(u);
  }
}
int RdP::localizar_indice_do_lugar(char desc[7])
```

```
{
  struct lugar *p=ME.lugares;
  int indice=0;
  if (!strcmp(p->descricao,desc))
    return indice;
  else
  {
    do
    {
      indice++;
      p=p->prox;
    }while (strcmp(p->descricao,desc));
    return indice;
  }
}
int RdP::localizar_indice_da_transicao(Transicao desc, char lo[7], char ld[7])
{
  struct transicao *p=ME.transicoes;
  int indice=0;
  if (p->descricao==desc && !strcmp(p->lugar_origem,lo) && !strcmp(p->lugar_destino,ld))
    return indice;
  else
  {
    do
    {
      indice++;
      p=p->prox;
    }while (p->descricao!=desc || strcmp(p->lugar_origem,lo) || strcmp(p->lugar_destino,ld));
    return indice;
  }
}
int RdP::localizar_posicao_de_insercao_lugar(char desc[7])
{
  lugar *p=ME.lugares;
  if (!p)
    return 1;
  else
  {
    if (!strcmp(p->descricao,desc))
      return 0;
    else
    {
      int posicao=2;
      while (p->prox && strcmp(p->prox->descricao,desc))
      {
        posicao++;
```

```
          p=p->prox;
        }
        if (p->prox && !strcmp(p->prox->descricao,desc))
          return 0;
        else
          return posicao;
      }
    }
}
void RdP::inserir_lugar(Inf_Frame frame, Situacao s)
{
   char desc[7];
   char time=frame.posse_bola_oponente()?'o':'t';
   int jogador=frame.consultar_jogador_com_posse_da_bola();
   int quadrante;
   if (s==gol)
   {
      if    ((frame.coordenada_y_bola()>=0    &&    frame.coordenada_x_bola()>0)    ||
(frame.coordenada_y_bola()<=0 && frame.coordenada_x_bola()<0))
        strcpy(desc,"gol_ld");
      else
        strcpy(desc,"gol_le");
   }
   else
   {
      if (s==bola_fora_l)
      {
        quadrante=frame.retorna_quadrante_bola_fora_lateral();
        sprintf(desc,"%d",quadrante);
      }
      else
        if (s==bola_fora_f)
        {
          quadrante=frame.retorna_quadrante_bola_fora_fundo();
          sprintf(desc,"%d",quadrante);
        }
        else
        {
          quadrante=frame.retorna_quadrante_bola();
          sprintf(desc,"%d%c%d",quadrante,time,jogador);
        }
   }
   int posicao=localizar_posicao_de_insercao_lugar(desc);
   if (posicao)
   {
      struct lugar *novo1,*novo2;
      novo1 = new struct lugar;
      novo2 = new struct lugar;
```

```
      strcpy(novo1->descricao,desc);
      novo1->PC=NULL;
      strcpy(novo2->descricao,desc);
      novo2->PC=NULL;
      if (posicao==1)
      {
        novo1->prox=ME.lugares;
        ME.lugares=novo1;
        novo2->prox=MS.lugares;
        MS.lugares=novo2;
      }
      else
      {
        struct lugar *aux1=ME.lugares, *aux2=MS.lugares;
        while (posicao>2)
        {
          aux1=aux1->prox;
          aux2=aux2->prox;
          posicao--;
        }
        novo1->prox=aux1->prox;
        aux1->prox=novo1;
        novo2->prox=aux2->prox;
        aux2->prox=novo2;
      }
    }
}
int RdP::localizar_posicao_de_insercao_transicao(Transicao desc, char lo[7], char
ld[7])
{
  transicao *p=ME.transicoes;
  if (!p)
    return 1;
  else
  {
    if (p->descricao==desc && !strcmp(p->lugar_origem,lo) && !strcmp(p-
>lugar_destino,ld))
      return 0;
    else
    {
      int posicao=2;
      while (p->prox && (p->prox->descricao!=desc || strcmp(p->prox-
>lugar_origem,lo) || strcmp(p->prox->lugar_destino,ld)))
      {
        posicao++;
        p=p->prox;
      }
```

```
        if (p->prox && p->prox->descricao==desc && !strcmp(p->prox-
>lugar_origem,lo) && !strcmp(p->prox->lugar_destino,ld))
          return 0;
        else
          return posicao;
      }
    }
}
void RdP::inserir_transicao(Transicao desc, Inf_Frame frame_LO, Inf_Frame
frame_LD, Situacao s)
{
  char LO[7];
  char time=frame_LO.posse_bola_oponente()?'o':'t';
  int jogador=frame_LO.consultar_jogador_com_posse_da_bola();
  int quadrante=frame_LO.retorna_quadrante_bola();
  sprintf(LO,"%d%c%d",quadrante,time,jogador);
  char LD[7];
  time=frame_LD.posse_bola_oponente()?'o':'t';
  jogador=frame_LD.consultar_jogador_com_posse_da_bola();
  if (s==gol)
  {
    if ((frame_LD.coordenada_y_bola()>=0 && frame_LD.coordenada_x_bola()>0) ||
(frame_LD.coordenada_y_bola()<=0 && frame_LD.coordenada_x_bola()<0))
      strcpy(LD,"gol_ld");
    else
      strcpy(LD,"gol_le");
  }
  else
  {
    if (s==bola_fora_l)
    {
      quadrante=frame_LD.retorna_quadrante_bola_fora_lateral();
      sprintf(LD,"%d",quadrante);
    }
    else
      if (s==bola_fora_f)
      {
        quadrante=frame_LD.retorna_quadrante_bola_fora_fundo();
        sprintf(LD,"%d",quadrante);
      }
      else
      {
        quadrante=frame_LD.retorna_quadrante_bola();
        sprintf(LD,"%d%c%d",quadrante,time,jogador);
      }
  }
  int posicao=localizar_posicao_de_insercao_transicao(desc, LO, LD);
  if (posicao)
```

```
{
   struct transicao *novo1,*novo2;
   novo1 = new struct transicao;
   novo2 = new struct transicao;
   novo1->descricao=desc;
   strcpy(novo1->lugar_origem,LO);
   strcpy(novo1->lugar_destino,LD);
   novo1->PL=NULL;
   novo2->descricao=desc;
   strcpy(novo2->lugar_origem,LO);
   strcpy(novo2->lugar_destino,LD);
   novo2->PL=NULL;
   if (posicao==1)
   {
      novo1->prox=ME.transicoes;
      ME.transicoes=novo1;
      novo2->prox=MS.transicoes;
      MS.transicoes=novo2;
   }
   else
   {
      struct transicao *aux1=ME.transicoes, *aux2=MS.transicoes;
      while (posicao>2)
      {
         aux1=aux1->prox;
         aux2=aux2->prox;
         posicao--;
      }
      novo1->prox=aux1->prox;
      aux1->prox=novo1;
      novo2->prox=aux2->prox;
      aux2->prox=novo2;
   }
}
//atualizando matrizes
int indice_LO=localizar_indice_do_lugar(LO);
int indice_LD=localizar_indice_do_lugar(LD);
int indice_T=localizar_indice_da_transicao(desc,LO,LD);
lugar *pl;
transicao *pt;
elemento *pe;
//atualizando matriz de entrada
pl=ME.lugares;
pt=ME.transicoes;
for (int i=0; i<indice_T; i++)
   pt=pt->prox;
for (int i=0; i<indice_LO; i++)
   pl=pl->prox;
```

```
if (!(pt->PL) || (pt->PL && pt->PL->j>indice_LO))
{
   struct elemento *novo1;
   novo1 = new struct elemento;
   novo1->valor=1;
   novo1->ocorrencia=1;
   novo1->i=indice_T;
   novo1->j=indice_LO;
   novo1->PL=pt->PL;
   pt->PL=novo1;
}
else
{
   if (pt->PL->j==indice_LO)
      (pt->PL->ocorrencia)++;
   else
   {
      pe=pt->PL;
      while (pe->PL && pe->PL->j<indice_LO)
         pe=pe->PL;
      if (!(pe->PL) || (pe->PL && pe->PL->j!=indice_LO))
      {
         struct elemento *novo1;
         novo1 = new struct elemento;
         novo1->valor=1;
         novo1->ocorrencia=1;
         novo1->i=indice_T;
         novo1->j=indice_LO;
         novo1->PL=pe->PL;
         pe->PL=novo1;
      }
      else
         (pe->PL->ocorrencia)++;
   }
}
if (!(pl->PC) || (pl->PC && pl->PC->i>indice_T))
{
   struct elemento *novo1;
   novo1 = new struct elemento;
   novo1->valor=1;
   novo1->ocorrencia=1;
   novo1->i=indice_T;
   novo1->j=indice_LO;
   novo1->PC=pl->PC;
   pl->PC=novo1;
}
else
{
```

```
    if (pl->PC->i==indice_T)
      (pl->PC->ocorrencia)++;
    else
    {
      pe=pl->PC;
      while (pe->PC && pe->PC->i<indice_T)
        pe=pe->PC;
      if (!(pe->PC) || (pe->PC && pe->PC->j!=indice_LO))
      {
        struct elemento *novo1;
        novo1 = new struct elemento;
        novo1->valor=1;
        novo1->ocorrencia=1;
        novo1->i=indice_T;
        novo1->j=indice_LO;
        novo1->PC=pe->PC;
        pe->PC=novo1;
      }
      else
        (pe->PC->ocorrencia)++;
    }
  }
  //atualizando matrzi de saida
  pl=MS.lugares;
  pt=MS.transicoes;
  for (int i=0; i<indice_T; i++)
    pt=pt->prox;
  for (int i=0; i<indice_LO; i++)
    pl=pl->prox;
  if (!(pt->PL) || (pt->PL && pt->PL->j>indice_LD))
  {
    struct elemento *novo2;
    novo2 = new struct elemento;
    novo2->valor=1;
    novo2->ocorrencia=1;
    novo2->i=indice_T;
    novo2->j=indice_LD;
    novo2->PL=pt->PL;
    pt->PL=novo2;
  }
  else
  {
    if (pt->PL->j==indice_LD)
      (pt->PL->ocorrencia)++;
    else
    {
      pe=pt->PL;
      while (pe->PL && pe->PL->j<indice_LD)
```

```
        pe=pe->PL;
      if (!(pe->PL) || (pe->PL && pe->PL->j!=indice_LD))
      {
        struct elemento *novo2;
        novo2 = new struct elemento;
        novo2->valor=1;
        novo2->ocorrencia=1;
        novo2->i=indice_T;
        novo2->j=indice_LD;
        novo2->PL=pe->PL;
        pe->PL=novo2;
      }
      else
        (pe->PL->ocorrencia)++;
    }
  }
  if (!(pl->PC) || (pl->PC && pl->PC->i>indice_T))
  {
    struct elemento *novo2;
    novo2 = new struct elemento;
    novo2->valor=1;
    novo2->ocorrencia=1;
    novo2->i=indice_T;
    novo2->j=indice_LD;
    novo2->PC=pl->PC;
    pl->PC=novo2;
  }
  else
  {
    if (pl->PC->i==indice_T)
      (pl->PC->ocorrencia)++;
    else
    {
      pe=pl->PC;
      while (pe->PC && pe->PC->i<indice_T)
        pe=pe->PC;
      if (!(pe->PC) || (pe->PC && pe->PC->j!=indice_LD))
      {
        struct elemento *novo2;
        novo2 = new struct elemento;
        novo2->valor=1;
        novo2->ocorrencia=1;
        novo2->i=indice_T;
        novo2->j=indice_LD;
        novo2->PC=pe->PC;
        pe->PC=novo2;
      }
      else
```

```
            (pe->PC->ocorrencia)++;
        }
    }
}
void RdP::escrever_RdP_em_arquivo (FILE *pf)
{
    lugar *p;
    transicao *q;
    elemento *r;
    //imprimindo a matriz de entrada
    p=ME.lugares;
    q=ME.transicoes;
    int nc=0;
    fprintf(pf,"\nMatriz de Entrada\n\t\t\t\t");
    while (p)
    {
        fprintf(pf,"\t%s",p->descricao);
        p=p->prox;
        nc++;
    }
    while (q)
    {
        fprintf(pf,"\n%s/%s\t%s",q->lugar_origem,q->lugar_destino,q-
>descricao==passe?"passe\t\t":(q->descricao==conducao?"conducao\t":(q-
>descricao==perda_posse_bola?"perda da posse de bola":"chute\t\t")));
        r=q->PL;
        for (int j=0; j<nc; j++)
        {
            if (r && r->j==j)
            {
                fprintf (pf,"\t%d",r->valor);
                r=r->PL;
            }
            else
                fprintf (pf,"\t0");
        }
        q=q->prox;
    }
    //imprimindo a matriz de saida
    p=MS.lugares;
    q=MS.transicoes;
    nc=0;
    fprintf(pf,"\n\n\nMatriz de Saida\n\t\t\t\t");
    while (p)
    {
        fprintf(pf,"\t%s",p->descricao);
        p=p->prox;
        nc++;
```

```
  }
  while (q)
  {
    fprintf(pf,"\n%s/%s\t%s",q->lugar_origem,q->lugar_destino,q-
>descricao==passe?"passe\t\t":(q->descricao==conducao?"conducao\t":(q-
>descricao==perda_posse_bola?"perda da posse de bola":"chute\t\t")));
    r=q->PL;
    for (int j=0; j<nc; j++)
    {
      if (r && r->j==j)
      {
        fprintf (pf,"\t%d",r->valor);
        r=r->PL;
      }
      else
        fprintf (pf,"\t0");
    }
    q=q->prox;
  }
}
void RdP::escrever_RdP_com_frequencia_em_arquivo (FILE *pf)
{
  lugar *p;
  transicao *q;
  elemento *r;
  //imprimindo a matriz de entrada
  p=ME.lugares;
  q=ME.transicoes;
  int nc=0;
  fprintf(pf,"\nMatriz de Entrada\n\t\t\t\t");
  while (p)
  {
    fprintf(pf,"\t%s",p->descricao);
    p=p->prox;
    nc++;
  }
  while (q)
  {
    fprintf(pf,"\n%s/%s\t%s",q->lugar_origem,q->lugar_destino,q-
>descricao==passe?"passe\t\t":(q->descricao==conducao?"conducao\t":(q-
>descricao==perda_posse_bola?"perda da posse de bola":"chute\t\t")));
    r=q->PL;
    for (int j=0; j<nc; j++)
    {
      if (r && r->j==j)
      {
        fprintf (pf,"\t%d",r->ocorrencia);
        r=r->PL;
```

```
        }
        else
          fprintf (pf,"\t0");
      }
      q=q->prox;
    }
    //imprimindo a matriz de saida
    p=MS.lugares;
    q=MS.transicoes;
    nc=0;
    fprintf(pf,"\n\n\nMatriz de Saida\n\t\t\t\t");
    while (p)
    {
      fprintf(pf,"\t%s",p->descricao);
      p=p->prox;
      nc++;
    }
    while (q)
    {
      fprintf(pf,"\n%s/%s\t%s",q->lugar_origem,q->lugar_destino,q-
>descricao==passe?"passe\t\t":(q->descricao==conducao?"conducao\t":(q-
>descricao==perda_posse_bola?"perda da posse de bola":"chute\t\t")));
      r=q->PL;
      for (int j=0; j<nc; j++)
      {
        if (r && r->j==j)
        {
          fprintf (pf,"\t%d",r->ocorrencia);
          r=r->PL;
        }
        else
          fprintf (pf,"\t0");
      }
      q=q->prox;
    }
}

class Construtor_RdP
{
    bool jogada_em_andamento;
    bool verificar_ocorrencia_de_inicio_de_jogada;
    bool verificar_finalizacao_de_jogada_por_perda_de_bola;
    bool analise_particular_verificar_finalizacao_de_jogada_por_perda_de_bola;
    bool analise_particular_verificar_ocorrencia_de_inicio_de_jogada;
    bool houve_conducao;
    Inf_Frame bkp_frame;
    Inf_Frame bkp_frame2;
    Inf_Frame origem_transicao;
```

```
  Historico historico;
public:
  Construtor_RdP(char *nome);
  ~Construtor_RdP();
  void atualizar_RdP(RdP &modelo_do_oponente, char  *vg);
};
Construtor_RdP::Construtor_RdP(char *nome)
{
  jogada_em_andamento=false;
  verificar_ocorrencia_de_inicio_de_jogada=false;
  verificar_finalizacao_de_jogada_por_perda_de_bola=false;
  analise_particular_verificar_finalizacao_de_jogada_por_perda_de_bola=false;
  analise_particular_verificar_ocorrencia_de_inicio_de_jogada=false;
  houve_conducao=false;
  historico.inicializar_historico(nome);
}
Construtor_RdP::~Construtor_RdP()
{
}
void Construtor_RdP::atualizar_RdP(RdP &modelo_do_oponente, char
*visao_global)
{
  int ocorreu;
  ocorreu=historico.atualizar_historico(visao_global);
  switch (ocorreu)
  {
    case 0:
    {
      if (verificar_ocorrencia_de_inicio_de_jogada)
        analise_particular_verificar_ocorrencia_de_inicio_de_jogada=true;
      if (verificar_finalizacao_de_jogada_por_perda_de_bola)
        analise_particular_verificar_finalizacao_de_jogada_por_perda_de_bola=true;
      break;
    }
    case 1:
    {
      verificar_ocorrencia_de_inicio_de_jogada=false;
      verificar_finalizacao_de_jogada_por_perda_de_bola=false;
      analise_particular_verificar_ocorrencia_de_inicio_de_jogada=false;
      analise_particular_verificar_finalizacao_de_jogada_por_perda_de_bola=false;
      houve_conducao=false;
      jogada_em_andamento=true;
      modelo_do_oponente.inserir_lugar(historico.retorna_frame(1), bola_em_jogo);
      origem_transicao=historico.retorna_frame(1);
      break;
    }
    case 2:
    {
```

```
        if (jogada_em_andamento)
        {
          verificar_finalizacao_de_jogada_por_perda_de_bola=true;
          analise_particular_verificar_finalizacao_de_jogada_por_perda_
de_bola=false;
          if      (historico.retorna_frame(3).retorna_quadrante_bola()!=historico.retorna_
frame(4).retorna_quadrante_bola())
          {
            houve_conducao=true;
            bkp_frame2=historico.retorna_frame(4);
          }
          bkp_frame=historico.retorna_frame(1);
        }
        break;
      }
      case 3:
      {
        if (verificar_ocorrencia_de_inicio_de_jogada)
        {
          modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
          origem_transicao=bkp_frame;
          analise_particular_verificar_ocorrencia_de_inicio_de_jogada=false;
          verificar_ocorrencia_de_inicio_de_jogada=false;
          jogada_em_andamento=true;
        }
        if (historico.consultar_ciclo()>0 && historico.consultar_ciclo()<80)
        if         (historico.retorna_frame(5).retorna_quadrante_bola()!=historico.retorna_
frame(6).retorna_quadrante_bola())
        {
          modelo_do_oponente.inserir_lugar(historico.retorna_frame(6),
bola_em_jogo);
          modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
historico.retorna_frame(6),bola_em_jogo);
          origem_transicao=historico.retorna_frame(6);
        }
        modelo_do_oponente.inserir_lugar(historico.retorna_frame(1), bola_em_jogo);
        modelo_do_oponente.inserir_transicao(passe,origem_transicao,
historico.retorna_frame(1),bola_em_jogo);
        origem_transicao=historico.retorna_frame(1);
        break;
      }
      case 4:
      {
        verificar_ocorrencia_de_inicio_de_jogada=false;
        verificar_finalizacao_de_jogada_por_perda_de_bola=false;
        analise_particular_verificar_ocorrencia_de_inicio_de_jogada=false;
        analise_particular_verificar_finalizacao_de_jogada_por_perda_de_bola=false;
        houve_conducao=false;
```

```
        jogada_em_andamento=false;
        break;
      }
      case 5:
      {
        if (verificar_finalizacao_de_jogada_por_perda_de_bola)
        {
          if (houve_conducao)
          {
            modelo_do_oponente.inserir_lugar(historico.retorna_frame(4),
bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
historico.retorna_frame(4),bola_em_jogo);
          }
        }
        else
          if (jogada_em_andamento)
            if                    (historico.retorna_frame(3).retorna_quadrante_bola()!=
historico.retorna_frame(4).retorna_quadrante_bola())
            {
              modelo_do_oponente.inserir_lugar(bkp_frame2, bola_em_jogo);
              modelo_do_oponente.inserir_transicao(conducao,
origem_transicao,bkp_frame2,bola_em_jogo);
            }
        break;
      }
      case 6:
      {
        if (analise_particular_verificar_finalizacao_de_jogada_por_perda_de_bola)
        {
          if (houve_conducao)
          {
            modelo_do_oponente.inserir_lugar(bkp_frame2, bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
bkp_frame2,bola_em_jogo);
            origem_transicao=bkp_frame2;
            houve_conducao=false;
          }
          modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
          modelo_do_oponente.inserir_transicao(perda_posse_bola,
origem_transicao,bkp_frame,bola_em_jogo);
          origem_transicao=bkp_frame;
          verificar_finalizacao_de_jogada_por_perda_de_bola=false;
          analise_particular_verificar_finalizacao_de_jogada_por_perda_de_
bola=false;
          jogada_em_andamento=false;
        }
        break;
```

```
    }
    case 7:
    {
      if (analise_particular_verificar_ocorrencia_de_inicio_de_jogada)
      {
        verificar_ocorrencia_de_inicio_de_jogada=false;
        verificar_finalizacao_de_jogada_por_perda_de_bola=false;
        analise_particular_verificar_ocorrencia_de_inicio_de_jogada=false;
        analise_particular_verificar_finalizacao_de_jogada_por_perda_de_
bola=false;
        houve_conducao=false;
        jogada_em_andamento=true;
        modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
        origem_transicao=bkp_frame;
      }
      break;
    }
    case 8:
    {
      if (verificar_ocorrencia_de_inicio_de_jogada)
      {
        modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
        origem_transicao=bkp_frame;
      }
      if (verificar_finalizacao_de_jogada_por_perda_de_bola)
      {
        historico.restaurar_historico();
      }
      if (historico.retorna_frame(3).retorna_quadrante_bola()!=historico.retorna_
frame(4).retorna_quadrante_bola())
      {
        modelo_do_oponente.inserir_lugar(historico.retorna_frame(4),
bola_em_jogo);
        modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
historico.retorna_frame(4),bola_em_jogo);
        origem_transicao=historico.retorna_frame(4);
      }
      modelo_do_oponente.inserir_lugar(historico.retorna_frame(1), gol);
      modelo_do_oponente.inserir_transicao(chute,origem_transicao,
historico.retorna_frame(1),gol);
      origem_transicao=historico.retorna_frame(2);
      break;
    }
    case 9:
    {
      if (verificar_finalizacao_de_jogada_por_perda_de_bola)
      {
        if (houve_conducao)
```

```
        {
          modelo_do_oponente.inserir_lugar(bkp_frame2, bola_em_jogo);
          modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
bkp_frame2,bola_em_jogo);
          origem_transicao=bkp_frame2;
        }
        modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
        modelo_do_oponente.inserir_transicao(perda_posse_bola,
origem_transicao,bkp_frame,bola_em_jogo);
        origem_transicao=bkp_frame;
      }
      break;
    }
    case 10:
    {
      if (verificar_finalizacao_de_jogada_por_perda_de_bola)
      {
        if              (historico.atualizar_informacoes_para_analise_da_ocorrencia_de_
conducao())
          houve_conducao=false;
        else
        {
          if (houve_conducao)
          {
            modelo_do_oponente.inserir_lugar(bkp_frame2, bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
bkp_frame2,bola_em_jogo);
            origem_transicao=bkp_frame2;
            houve_conducao=false;
          }
          modelo_do_oponente.inserir_lugar(historico.retorna_frame(1),
bola_em_jogo);
          modelo_do_oponente.inserir_transicao(passe,origem_transicao,
historico.retorna_frame(1),bola_em_jogo);
          origem_transicao=historico.retorna_frame(1);
        }
        verificar_finalizacao_de_jogada_por_perda_de_bola=false;
        analise_particular_verificar_finalizacao_de_jogada_por_perda_de_
bola=false;
      }
      else
      {
        verificar_ocorrencia_de_inicio_de_jogada=true;
        analise_particular_verificar_ocorrencia_de_inicio_de_jogada=false;
        bkp_frame=historico.retorna_frame(1);
      }
      break;
    }
```

```
    case 11:
    {
      if (jogada_em_andamento)
      {
        if (verificar_finalizacao_de_jogada_por_perda_de_bola)
        {
          if (houve_conducao)
          {
            modelo_do_oponente.inserir_lugar(bkp_frame2, bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
bkp_frame2,bola_em_jogo);
            origem_transicao=bkp_frame2;
          }
          modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
          modelo_do_oponente.inserir_transicao(perda_posse_bola,
origem_transicao,bkp_frame,bola_em_jogo);
          origem_transicao=bkp_frame;
        }
        else
        {
          if    (historico.retorna_frame(3).retorna_quadrante_bola()!=historico.retorna_
frame(4).retorna_quadrante_bola())
          {
            modelo_do_oponente.inserir_lugar(historico.retorna_frame(4),
bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
historico.retorna_frame(4),bola_em_jogo);
            origem_transicao=historico.retorna_frame(4);
          }
          modelo_do_oponente.inserir_lugar(historico.retorna_frame(2), bola_fora_l);
          modelo_do_oponente.inserir_transicao(passe,origem_transicao,
historico.retorna_frame(2),bola_fora_l);
          origem_transicao=historico.retorna_frame(2);
        }
      }
      break;
    }
    case 12:
    {
      if (jogada_em_andamento)
      {
        if (verificar_finalizacao_de_jogada_por_perda_de_bola)
        {
          if (houve_conducao)
          {
            modelo_do_oponente.inserir_lugar(bkp_frame2, bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
bkp_frame2,bola_em_jogo);
```

```
            origem_transicao=bkp_frame2;
          }
          modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
          modelo_do_oponente.inserir_transicao(perda_posse_bola,
origem_transicao,bkp_frame,bola_em_jogo);
          origem_transicao=bkp_frame;
        }
        else
        {
          if    (historico.retorna_frame(3).retorna_quadrante_bola()!=historico.retorna_
frame(4).retorna_quadrante_bola())
          {
            modelo_do_oponente.inserir_lugar(historico.retorna_frame(4),
bola_em_jogo);
            modelo_do_oponente.inserir_transicao(conducao,origem_transicao,
historico.retorna_frame(4),bola_em_jogo);
            origem_transicao=historico.retorna_frame(4);
          }
          modelo_do_oponente.inserir_lugar(historico.retorna_frame(2), bola_fora_f);
          modelo_do_oponente.inserir_transicao(chute,origem_transicao,
historico.retorna_frame(2),bola_fora_f);
          origem_transicao=historico.retorna_frame(2);
        }
      }
      else
      {
        if (verificar_ocorrencia_de_inicio_de_jogada)
        {
          modelo_do_oponente.inserir_lugar(bkp_frame, bola_em_jogo);
          origem_transicao=bkp_frame;
          modelo_do_oponente.inserir_lugar(historico.retorna_frame(2), bola_fora_f);
          modelo_do_oponente.inserir_transicao(chute,origem_transicao,
historico.retorna_frame(2),bola_fora_f);
          origem_transicao=historico.retorna_frame(2);
        }
      }
      break;
    }
  }
}
```