STEP3D-05 Team Description

Alexey Krichun[1], Sergey Serebryakov[2], Lev Stankevitch[1]

[1]"RosElectroProm Holding" Company
7, ul. Elektropultovtsev, St. Petersburg, 195030, Russia
alexy_kr@mail.ru, stankevich_lev@inbox.ru
[2] Saint Petersburg Institute for Informatics and Automation of the
Russian Academy of Science
39, 14-th liniya V.O., St. Petersburg, 199178, Russia
sergey_s@iias.spb.su

Abstract. STEP3D-05 is a new 3D Soccer Simulation League team being developed by the members of 2D League team STEP development group. In this paper we describe basic design principles of STEP3D agent.

Introduction

STEP3D-05 is a new 3D Soccer Simulation League team being developed by the members of 2D League team STEP development group at RosElectroprom Holding in cooperation with St. Petersburg Institute for Informatics and Automation of Russian Academy of Sciences, and students of St. Petersburg State Polytechnical University. The team code development is based on previous 3-year experience in 2D team development, and we tried to use similar design principles as in our 2D agent.

Agent structure

Agent behavior implementation

Agent behavior structure is shown in Fig. 1. Each block represents respective class of the agent. Each class has pointer to agent world model. This structure is developed for future researches, so many of its blocks are yet not implemented for 3D agent.

- **ServerCommands.** Commands, produced by server. It's a heritable class from Netutils, which is responsible for network operations and for communication with communication server.
- **Kick.** Simple kick routines. Kick is done in the direction of line, which is from agent body's center to ball's center. So, x-y kick angle can't be changed.
- **Drive.** Drive routines. Driving to a particular position or direction. Stop skill is also here.

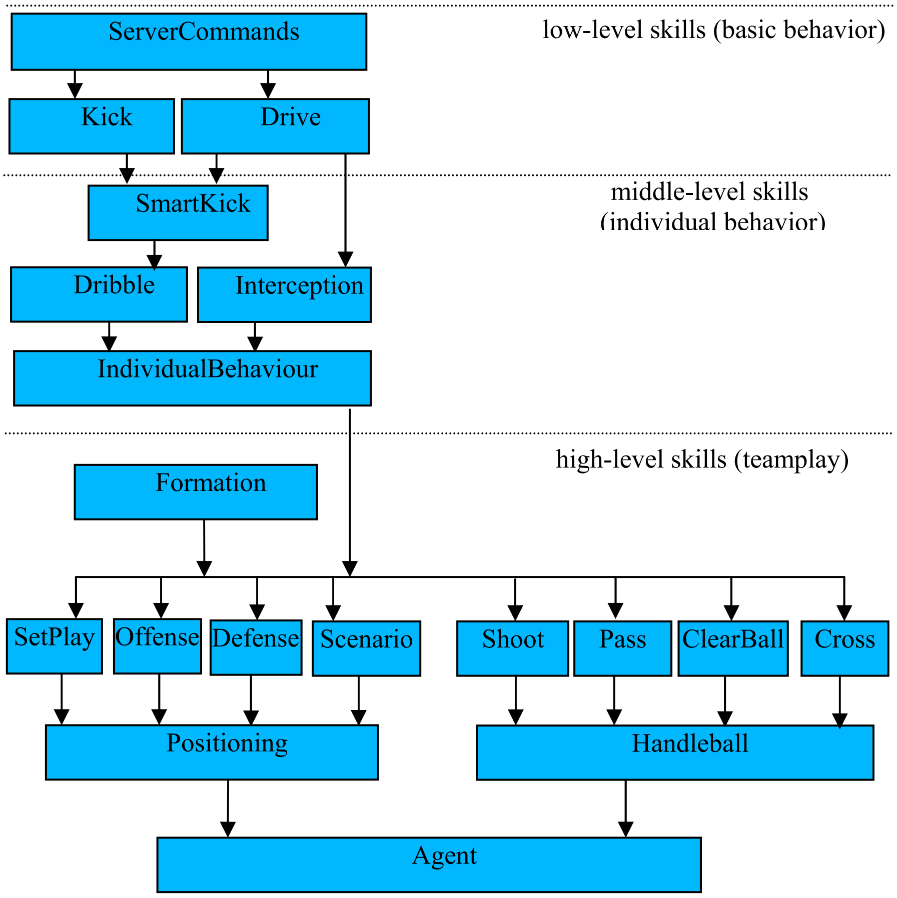

Fig. 1. Agent behavior structure

- **SmartKick.** Smart kick routines, kicking ball in any direction. To do it, agent needs to adjust its position, i.e. to move to a certain pos close to the ball. This skill is allowed to move the player, so even if the ball is not kickable, this skill will move the player to a position, close to the ball.
- **Dribble**. Dribble ball to a specific position or direction.
- **Interception**. Interception skill.
- **IndividualBehaviour**. Individual behavior just inherits 2 above classes and adds new stuff, such as confidence of kicking from point to point, which is widely used through the agent.
- **Formation**. Team formation.
- **SetPlay**. Set play behavior. What to do in different standard situations.
- **Offense**. Rules for offense behavior (fast attack, open for pass ...).
- **Defense**. Rules for defense behavior (marking, pressing ...).
- **Scenario**. Scenario team play.
- **Positioning**. Responsible for behavior without ball (inherits previous behaviors).

- **Shoot**. Shoot skill (try to score the goal).
- **Pass**. Pass skill (pass ball to teammates).
- **ClearBall**. Clear ball skill (kick the ball such in a way to have opponents as much time as possible to intercept the ball).
- **Cross**. Cross skill (crossing ball close to opponent's penalty area for goalie outplaying).
- **Handleball**. Responsible for with-ball behavior (inherits above behaviors and makes top level decisions).
- **Agent**. Overall agent behavior. It contains goalie behavior as well.

World model

World model of the agent consists of 4 classes (this structure is similar to world model used in CMUnited-99 team, which we used in our 2D team). World model structure is shown in Fig. 2.

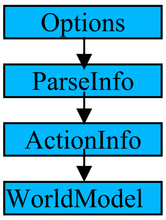

Fig. 2. World model

Basic class is Options class. It contains global options used by agent and all its parts. ParseInfo class contains information parsed from server messages. ActionInfo contains information about agent's actions. WorldModel inherits 3 above classes, and adds information about current situation.

Localization

For accurate estimation of current position, our agent tries to filter out noise server adds to agent's vision. Agent takes noised flag position from vision sensor and then uses Kalman filtering to estimate real position.

Formations

Team formation structure of the agent is similar to 2D agent. Each team member has its own role depending on its uniform number: stopper, defender, midfielder, attacker and goalie. Every player has field zone assigned to it: center, left or right.

Home position of the player linearly depends on ball position:

$$\begin{aligned} \mathrm{hom}\, e\_x_i &= base_{x,i} + alpha_{x,i} * ball\_x \\ \mathrm{hom}\, e\_y_i &= base_{y,i} + alpha_{y,i} * ball\_y \end{aligned} \qquad \textbf{(1)}$$

where $base_{\{x,y\}I}$ and $alpha_{\{x,y\}I}$ are pre-defined formation parameters for i-th player, $1 \le i \le 11$.

This method has a disadvantage for defender position. When ball moves from field center to player's goal, home position changes too fast and defenders "fall" back to their goal. To avoid this, position calculation uses piecewise-linear model for fast calculation of all positions of the players:

$$\begin{aligned} & start_0 = -field\_length/2 \\ & start_{j+1} = \frac{base_{j+1} - base_j}{alpha_{j+1} - alpha_j} \\ & 0 \le j < num\_fracture \end{aligned} \qquad \textbf{(2)}$$

where *num_fracture* is number of linear pieces, *field_length* is length of the field.

Home position is calculated as follows:

$$\begin{aligned} \mathrm{hom} e\_x_{i,j} &= base_{x,i,j} + alpha_{x,i,j} * ball\_x \\ \mathrm{hom}\, e\_y_{i,j} &= base_{y,i,j} + alpha_{y,i,j} * ball\_y \end{aligned} \qquad \textbf{(3)}$$

where j is chosen as maximum when $start_j \le ball\_x$ (or $ball\_y$) $0 \le j < num\_fracture$.

This method helps defender to avoid falling back too fast.

Conclusion

Basic design principles of STEP3D team were described. As this team development is recently started, there are some problems to solve, such as good movement and shoot skills functions, advanced multiagent strategies and scenario implementation. We intend to use previously developed modules from our successful 2D agent together with newly developed cognitive modules for these functions.