CPSC 526 : Computer Animation

Term Project

"Using Fuzzy Logic to Control the Behaviour of a Hockey Defenseman Character"

Erwin Tang
83458927

December 14, 2004

1 INTRODUCTION

The demand for increased realism in video games grows year after year. Hockey video games in particular are a game genre that relies on realism for a game to be fun and accepted by the game-playing community. Because many people play hockey and watch it on TV, hockey video games are held to a high standard.

A large factor in providing realism in hockey games is how the non-player characters behave during gameplay. These are characters that are not controlled by the user. Are they in the right position at the right time? Do they react as they should to certain game situations? Behaviour which deviates from the expected detracts from the overall experience of playing the game.

In this report, I describe the use of fuzzy logic to control the behaviour of a non-player defenseman character. I chose to focus the control on the defenseman because in hockey, this role requires the strictest adherence to proper positioning. A good defenseman is always in the right position on the ice and knows when to change positions at the proper skating speed. This makes it easier for us to describe how a good defenseman should react.

I chose to use fuzzy logic as the control scheme because it has been shown to be applicable to video games [1] and existing development tools are already in place [2], [3]. Fuzzy logic was also chosen because it can easily translate real-world verbal descriptions into a control strategy. Most control strategies require equations which describe how one or many state variables change due to inputs. These equations need to be precise. Fuzzy logic control however, can be derived from verbal descriptions, such as “if the puck is in this zone, skate here”. Fuzzy logic goes beyond “crisp” models of AI and provides a much more realistic representation of the world.

1.1 RELATED WORK

The video game industry is filled with instances of proprietary code and a general desire to protect intellectual property. Unlike academia, video game developers are less open to share what novel ideas they have come up with. As such, it was difficult to discover what type of approaches have been tried to control the behaviour of video game characters. It is known that fuzzy logic has been used in video games in the past [1].

Listening to a classmate who has worked in the industry, he spoke of hockey player characters being controlled by a long series of "if-then" statements (without fuzzy control). As he described it, the approach appears simple but good enough for the game producers at this point.

1.2 OVERVIEW

Fig. 1 below shows an overview of the system that was developed. The user interface allows the user to set the state variables for the game and defenseman. Game state variables are the game time, game score, puck possession, and puck position. For the purposes of fuzzy control the defenseman only has a single state variable, its position. Note that there can be many sets of defenseman state variables in the simulation (representing different defensemen), but there can only be a single set of game state variables.

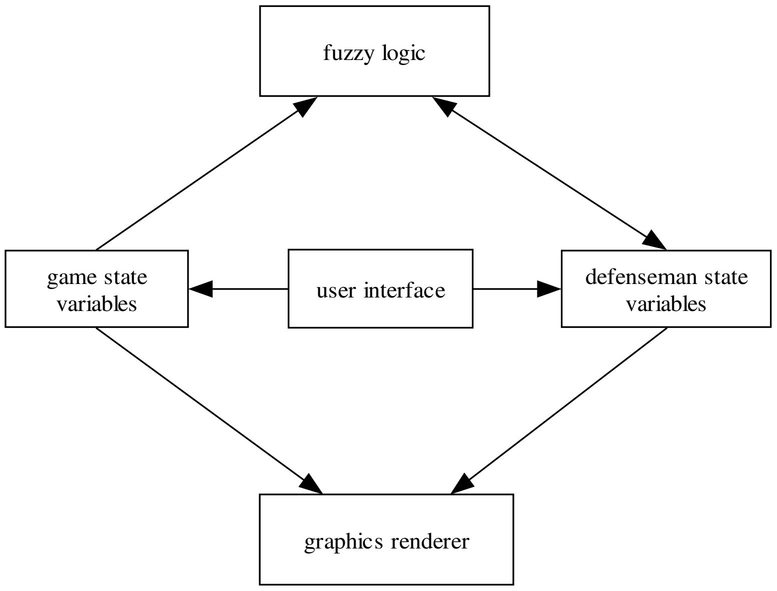

Fig. 1: System overview

During each tick of the simulation, the state variables are fed into the fuzzy logic system as inputs. The fuzzy logic system then calculates an updated "desired position" for the defenseman. Based on the difference between the defenseman's current position and the desired position, the fuzzy logic then calculates the velocity of the defenseman. Depending on this calculated velocity, the simulation will then set the new position of the defenseman for the next frame of animation. The higher the calculated velocity the farther the defenseman object will be drawn from its last position.

The fuzzy logic system that is implemented comes courtesy of the Free Fuzzy Logic Library API [3], which is an open-source implementation of a fuzzy logic engine.

Note that for the purposes of this simulation we will only be controlling the position of player along the length of the ice using fuzzy logic. For a defenseman, overall positioning along the length of the ice is much more important than across. Nonetheless, a simply but non-fuzzy system is implemented to control cross-ice movement.

2 FUZZY LOGIC PRIMER

To fully understand the system being discussed, one must understand a bit about fuzzy logic. Let us quickly review the most relevant fuzzy logic concepts.

Traditional logic deals with “crisp” sets. An element in the universe of discourse is either in a given set or it is not. For example, for all elements of non-zero numbers, they are either in the set of positive numbers or they are not. For “crisp” logic, one cannot have a number that is partially positive and partially negative. For fuzzy logic, however, such a description is possible.

Usually, how much an element belongs to a set is called its membership to that set. A function that describes this is called the membership function. Values of membership range from 0.0 to 1.0, inclusive, with 1.0 meaning an element totally belonging to that set. Membership functions have a particular shape as well. The most common shape is triangular or trapezoidal, but other shapes can be used as well.

Suppose that we consider having $200 belongs to set of “insanely rich” people. If you have $100, you are only “wealthy”. Finally, as you approach $0, you are considered “poor”. In Fig. 2 then, a person with $67.50 belongs to the “wealthy” set by 0.35 and the “poor” set by 0.10.

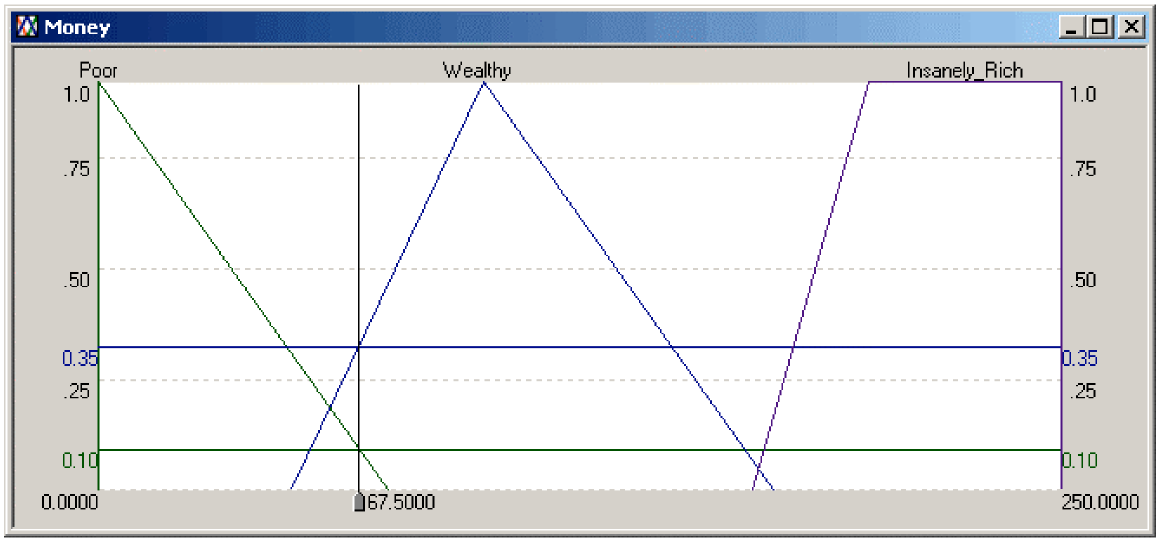

Fig. 2: Membership sets in a linguistic variable

The figure above could easily describe what is termed a *linguistic variable* in fuzzy logic. In this case, we could say the three sets are part of the "Economic Class" variable. The way we divide a linguistic variable into sets of membership functions is called *fuzzification*.

Once we have fuzzified input and output variables, we describe the interaction between inputs and the output with a series of *if-then* statements called *rules*. For a given set of inputs, more than one rule may come into play, each calling for a different type of action. The method in which this is resolved is called *defuzzification*. This allows the output to take on a crisp value. Defuzzification will be elaborated on further in the next few sections.

3 IMPLEMENTATION

There were four steps to implementing the system: fuzzifying the game and defenseman state variables, generating the rules, writing the FCL files and using the FFLL API, and integration with the graphics renderer.

3.1 FUZZIFYING THE GAME AND DEFENSEMAN STATE VARIABLES

It was decided that the fuzzy logic system would determine two outputs for the simulation: the desired position and the velocity for the defenseman. The inputs and outputs of the system are summarized below in Fig. 3.

Output	**Inputs**
Desired_Position	Puck_Position
	Possession
	Game_Score
	Game_Time

Output	**Input**
Velocity	Separation

Fig. 3: Inputs and outputs for the system

Note that the defenseman’s position is not explicitly included in the list of inputs anywhere. Instead, the player position is taken into consideration in the *separation* variable:

$$\textit{separation} = \textit{puck\_position} - \textit{player\_position}$$

The act of fuzzifying variables is almost as much art as it is science. A good first attempt will eliminate tweaking once the system is in place, but adjustments can be made later until the desired response is seen.

For the variable *Puck_Position*, the range of values consisted of -100 to +100. This represents the position of the puck along the length of the ice surface for a standard 200 ft NHL-size rink. Within this variable, seven sets were used to fuzzify the range:

DDEF = the deep part of the defensive zone
SDEF = the shallow part of the defensive zone
DNEU = the defensive part of the neutral zone
NEU = the middle part of the neutral zone
ONEU = the offensive part of the neutral zone
SOFF = the shallow part of the offensive zone
DOFF = the deep part of the offensive zone

Fig. 4 shows the fuzzified variable.

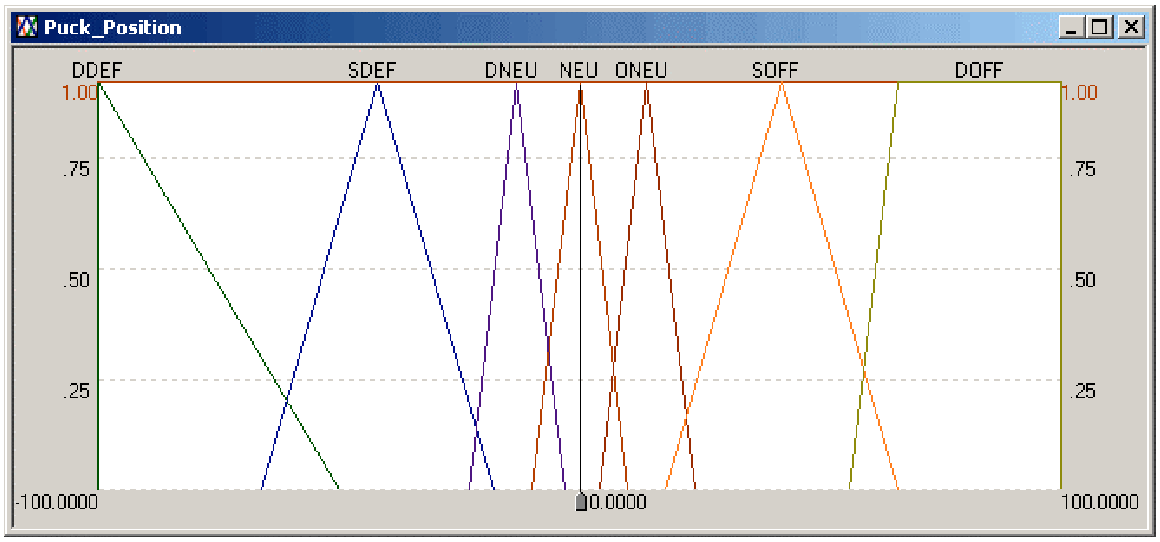

Fig. 4: Variable *Puck_Position*

It only makes sense to fuzzify the variable *Desired_Position* in the same way. After all, why use two different ice surface representations?

For *Possession* we use singleton values to represent the sets. Here we have a case of using fuzzy logic to represent crisp values. There is no partial membership in sets for this variable. *Possession* is 0 when no team has the puck, 1 when the user's team has the puck, and -1 when the opponent's team has the puck. See Fig. 5 below.

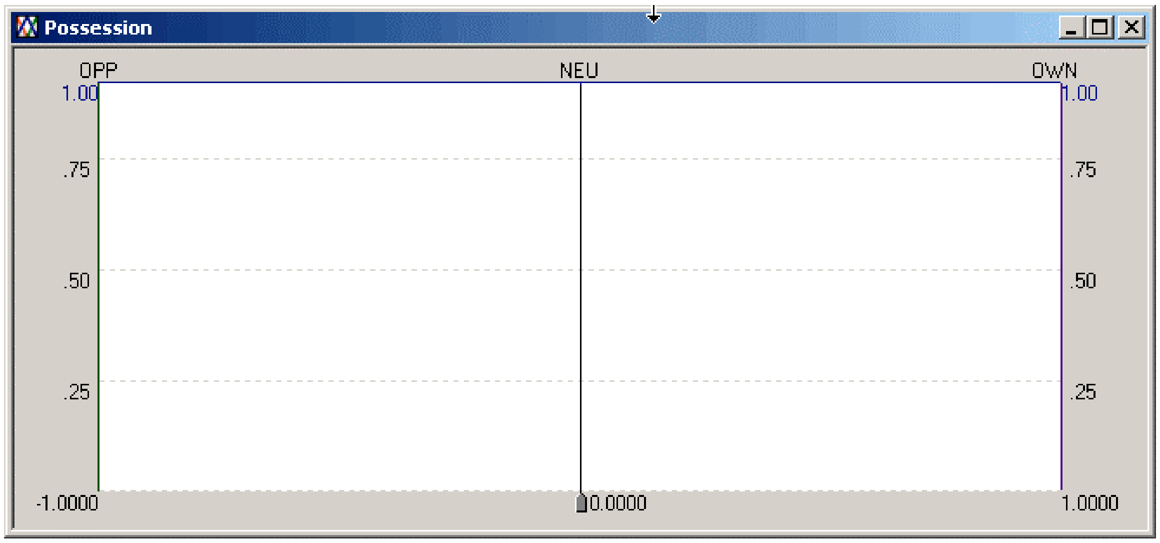

Fig. 5: Variable *Possession*

For *Game_Score*, this represents the integer difference between the user's team and the opponents team. Any negative value is assigned a 1.0 membership to the set BEH (for behind). When the difference is positive, membership fully belongs to the set AHE (for ahead). When the score is tied, membership fully belongs to TIE (for tied). See Fig. 6 below.

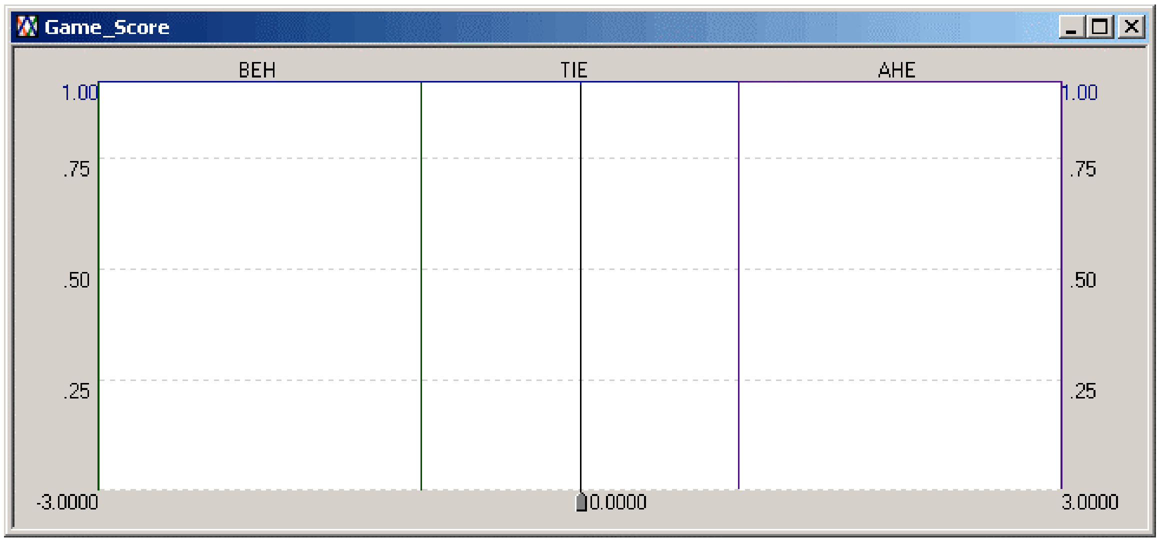

Fig. 6: Variable *Game_Score*

For *Game_Time*, this represents the number of minutes expired in a 60 minute hockey game. Games are divided into 20 minute periods for a total of three periods. There is a special set named *END* which begins its influence with about 10 minutes left in the game. This allows for special behaviour to come into play near the end of a game. See Fig. 7 below.

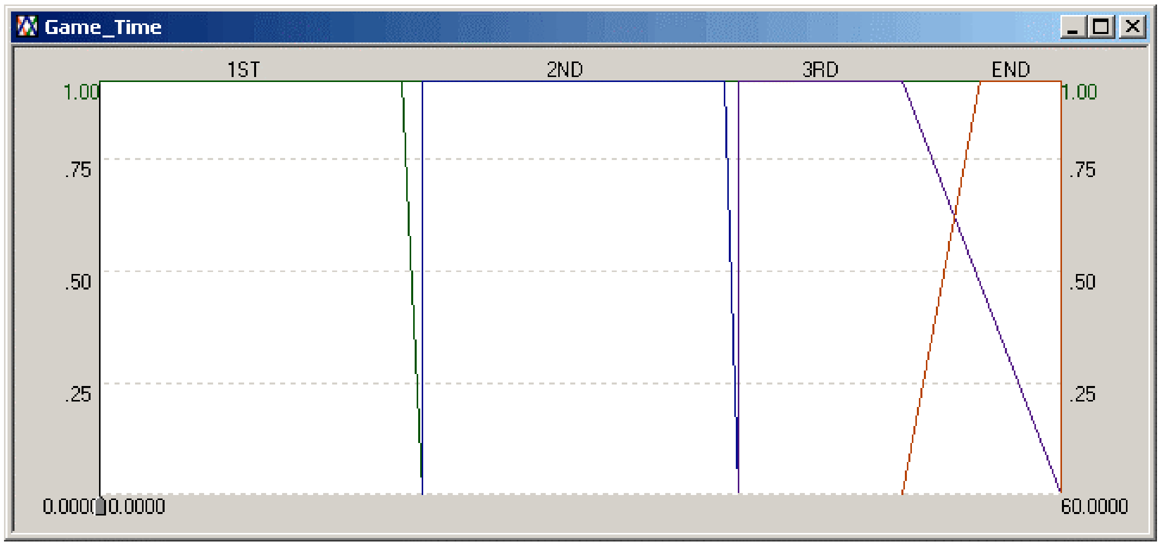

Fig. 7: Variable *Game_Time*

For the output variable *Velocity*, the range of values are from -29.3 ft/s to +29.3 ft/s. This is based on the top skating speed of 20 mph for an NHL player [4]. The velocity is positive when skating towards the offensive zone and negative when skating back to the defensive zone. See Fig. 8 below.

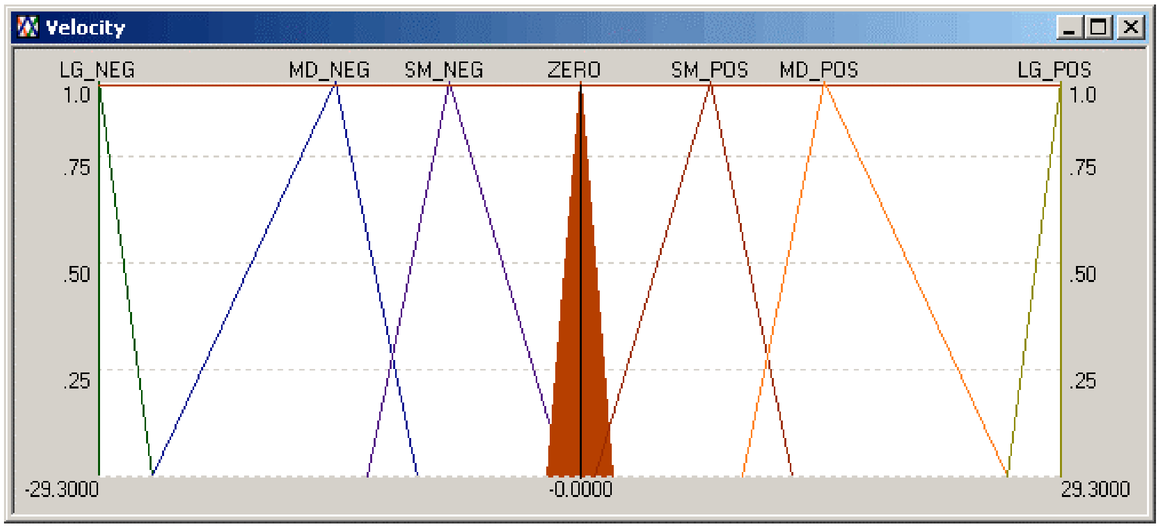

Fig. 8: Variable *Velocity*

The input that determines *Velocity* is the variable *Separation*. This variable was defined previously and represents the distance between the defenseman and the puck. A positive value means the puck is towards the offensive zone (in front of the defenseman) and a negative value means the puck is towards the defensive zone (behind the defenseman). Though the labels are hard to see in Fig. 9, they are the exact same sets used in *Velocity*.

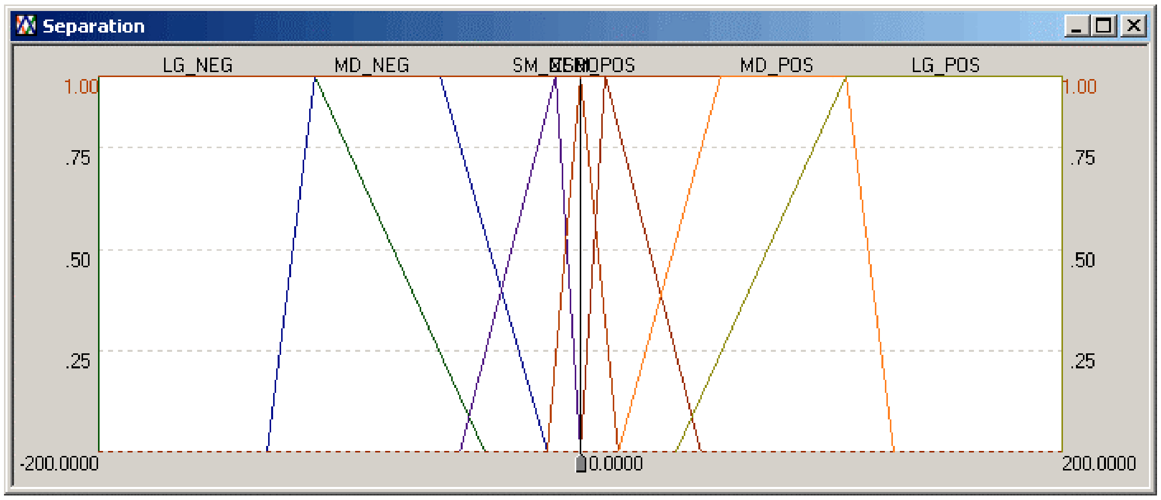

Fig. 9: Variable *Separation*

3.2 GENERATING THE RULES

The next step is to write the rules which govern the desired behaviour of the outputs for the system. As mentioned previously, these are a series of *if-then* statements. Note that if one neglects to write a rule for a particular set of inputs which can occur, the fuzzy logic system will not know how to respond. That is, there will be no value for the output in that case. This is why it is extremely important to write rules which cover every possible set of inputs.

Unfortunately, due to current limitations in the Free Fuzzy Logic Library (FFLL) API, it cannot accept fuzzy rules with OR statements. That is, it is not possible to write the following rule:

```
IF (var_a IS HOT OR WARM) THEN (var_b IS OPEN)
```

Instead, we must write two statements which together, are equivalent to the above:

```
IF (var_a IS HOT) THEN (var_b IS OPEN)
IF (var_a IS WARM) THEN (var_b IS OPEN)
```

One can easily see the problem faced without the use of OR statements. For the variable *Desired_Position*, we require 7*3*3*4 = 252 rules to completely describe the set of all possible inputs. It would tedious and prone to error if all 252 had to be written by hand. To solve this problem, a rule generation program called *rulegen* was written to help generate rules quickly. This program allows the user to hold a variable steady at a particular value while iterating through the values in the other variables. Using *rulegen*, all 252 rules were written in a matter of minutes.

The rules for *Velocity* were much simpler to write. Because the input *Separation* had only seven sets in it, only seven rules had to be written.

It cannot be stressed enough how important it is to write good rules that reflect the desired behaviour of the output. Together with the way we fuzzify the variables, the rules determine exactly how our defenseman will react in game situations.

Keeping this in mind, I wrote two sets of rules for *Desired_Position*. The first set describes what is known as a “stay-at-home” defenseman. This type of player plays a more conservative role in defense, staying back more than joining the play on offense. The second set of rules describes a more offensively-minded player, who knows their role on defense, but stays closer to the puck in all regions of the ice. This player is not afraid to stay in the offensive zone in more situations.

For each set of rules, special behaviour kicks in at the end of the game. Depending if one team is winning or losing, the behaviour of the defenseman changes. If the defenseman’s team is losing, then he will exhibit more offensive behaviour. He will stay in the offensive zone in situations that might be risky from a defensive point of view, but worth it to tie up the game.

If the defenseman’s team is ahead, his behaviour will become more conservative. Rarely will he venture into the offensive zone, instead, he will adopt a more defensive positional posture.

The entire set of rules used in the simulation maybe found in Appendix A which contain the FCL files.

3.3 WRITING THE FCL FILES AND USING THE FFLL API

The FFLL API mentioned earlier takes as input something called Fuzzy Control Language (FCL) files. FCL files are a somewhat standard protocol for describing fuzzy logic control systems [5]. They are essentially text files which describe the linguistic variables (input and output) and the way they have been fuzzified. The rules for the system are also listed.

While the FCL standard is quite rich, unfortunately, the FFLL API only supports a very small subset of the language. As such, some of the more useful options (for example, OR statements) are not available.

As mentioned previously, the FCL files for this system can be found in Appendix A. There are two useful benefits to using FCL files in an application. First, the behaviour of the fuzzy logic system can be altered by simply editing the FCL files in a text editor. No re-compilation of code is necessary. Since FCL files are relatively easy to understand, this gives the power to change the behaviour to someone that isn't necessarily a programmer. In a game development environment, this gives a game designer much more direct control on character behaviour.

The second benefit of using FCL files is that they can be read in by a free viewer called Spark! [6]. Spark! uses a graphical interface to display all the inputs and outputs for a fuzzy logic system written in FCL. A user can use sliders to vary the inputs to the system and immediately see the resulting defuzzified output value. Because of this, we can assess the performance of a fuzzy logic system without running a single line of user code.

Consider Fig. 10 below which shows the output *Desired_Position* for a particular set of inputs.

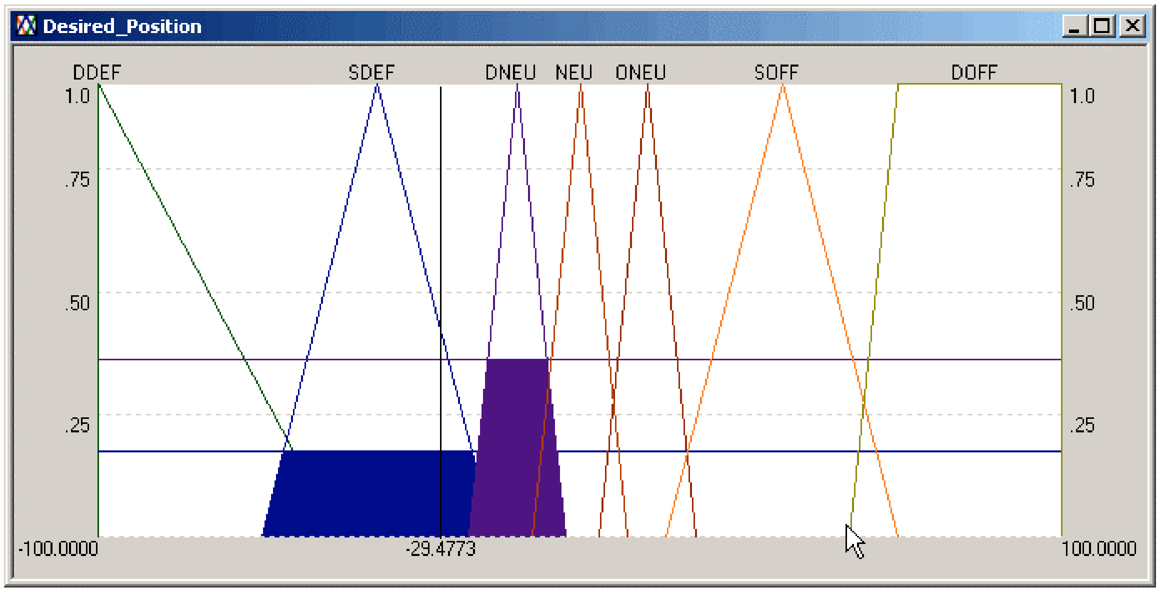

Fig. 10: Defuzzifying an output

We can see that the output is almost 0.25 for the SDEF zone and about 0.38 for the DNEU zone. Defuzzification determines exactly where the desired position should be. The shaded area in the figure describes the contribution of each set to the output. Defuzzifying the output involves finding the center of area for the shaded region. In this case, that value lies at around -29 feet behind the center line.

A simple call to the FFLL API will open any user-specified FCL file and initialize the fuzzy logic system for that file. The returned object is called a *model*. Each model can then spawn multiple *children*. One could then attach one child for every game object, so that multiple objects could use the same fuzzy logic model.

Inputs can be fed into the model at any time. Another call to the API will return the output for the model.

3.4 INTEGRATION WITH THE GRAPHICS RENDERER

For this project, both OpenGL and GLUT (OpenGL Utility Toolkit) were used to create graphics for the simulation.

The ice surface markings were drawn with OpenGL primitives. The defensemen are represented as simple green OpenGL points. Smaller points highlight each defenseman so that one can distinguish between players. Teammates are also drawn the same way, but without the highlight. Opposition players are drawn in the same fashion but with a different colour (yellow). The puck is also represented as a black OpenGL point.

A set of indicators to the right of the ice surface indicate which team has possession of the puck. For each team, an icon is displayed. When a team has the puck, a box is drawn around the icon. When no team has the puck no box is drawn. See Fig. 11 below.

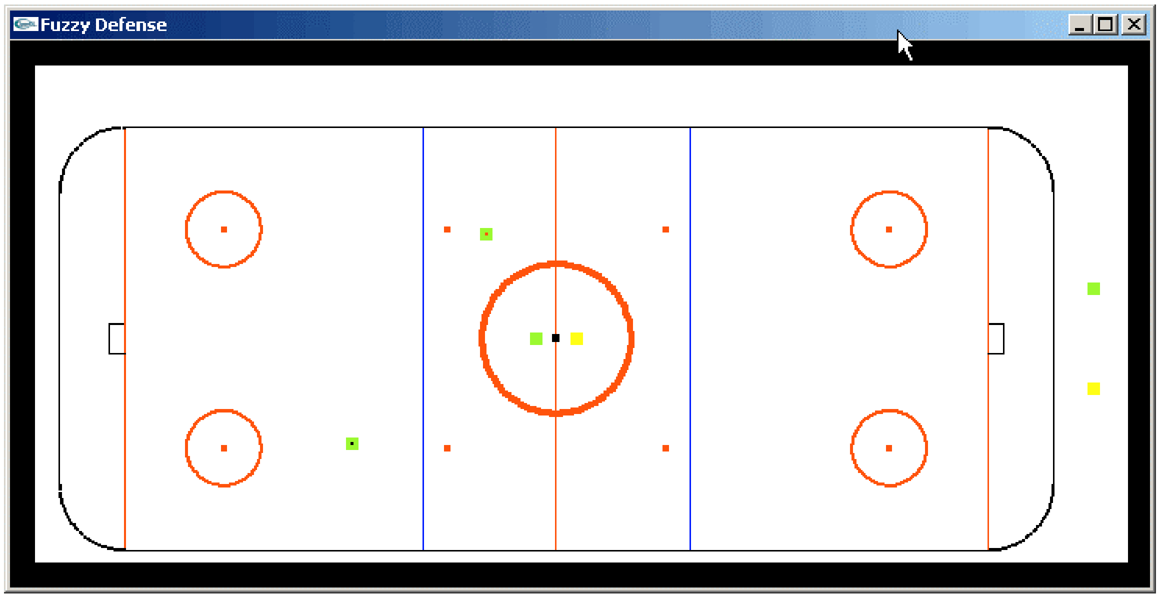

Fig. 11: Simulation screen capture

Every 33 milliseconds, a fuzzy logic system updates the desired positions and velocities of the defensemen. After the updates occur, a new frame in the simulation is drawn.

The user can adjust the positions of any on-ice object by clicking and dragging defensemen, teammates, opponents, and the puck. Users can also use the keyboard to adjust the score and game time. Please refer to Appendix B for a full list of commands.

4 TESTING AND RESULTS

For the purposes of testing, the following game situation was developed. The simulation had two defenseman, a teammate, an opponent, and the puck on the ice. The following scenarios were tested.

Reacting to the puck moving, no possession

This test consisted of dragging the puck along the ice surface with no team in possession of it. This corresponded to variable *Possession* set to NEU.

The first observation was the defenseman did not seem to be moving as fast as they should have been based on the separation distance. Using the Spark! viewer, it was discovered the maximum velocity defuzzified by the configuration of the membership functions was several ft/s below the 29.3 ft/s theoretical maximum. The shape of the membership functions for *Velocity* and *Separation* had to be tuned to achieve a greater calculated velocity. A good understanding of the center of area defuzzification calculation can lead to a much faster tuning of membership functions. Results of tuning can be seen immediately using the Spark! viewer.

The second observation was that while the defensemen went to right general area of the ice based on the puck position, they were sometimes a few feet too close or too far away from the ideal position. This led to further tuning of the shapes of the membership functions for variables *Desired_Position* and *Puck_Position*. Once tuned the defensemen went to the proper positions.

Reacting to the teammate with the puck

When the teammate moved with the puck, both defensemen reacted as specified by their rules. The more offensively-minded player stayed closer to the puck and stayed longer in the offensive zone. The more defensive-minded player followed his teammate at a farther distance.

Reacting to the opponent with the puck

When the opponent moved with the puck, both defensemen reacted as specified by their rules. The more aggressive player stayed closer to the puck as it traveled through the neutral zone, while the more conservative player backed up quicker and waited for the puck in the defensive zone.

Reacting to end of game, player’s team winning

In this situation, the player’s team is winning near the end of the game. As stated by the rules, both defenseman adopt a more defensive mindset. They are more prone to staying in the defensive zone and wait for the opponent to come to them. Only when their team has the puck do they venture out of the defensive zone.

Reacting to end of game, player’s team losing

In this situation, the player’s team is losing near the end of the game. As stated by the rules, both defenseman adopt a more offensive mindset in hopes of tying the game. Instead of vacating the offensive zone when the opposition has the puck or the puck is loose, they stay in the zone to help their forwards recover the puck. Also when the opponent moves with the puck, both defensemen will follow much closer in hopes of creating a turnover in possession.

5 CONCLUSIONS

As shown by the results of this project, it is possible to control the behaviour of video game hockey characters by using fuzzy logic.

One cannot overstate the importance of having sufficient rules to cover all the expected input states. If a rule has not been written for a particular set of states, the character will not know what to do. Also, having the correct rules and properly formed membership functions can save significant development time. Fortunately, tuning and tweaking of such parameters can be done until the desired results are achieved.

One possible drawback of using fuzzy logic control is that during the design process, it can be difficult to know what set of inputs will give a specific output. For example, say one must have a certain numerical output for a particular rule. It is difficult to know what shape to make the membership functions for the inputs to guarantee that particular output.

The use of FCL files and the FFLL API is overall a great convenience to the programmer. With API calls, the fuzzification and defuzzification process is abstracted away so that the programmer need not worry about those details. The use of FCL files means that a non-programmer can design the behaviour of the characters. This frees the task of AI design from the programmer and gives it to anyone who wants to do it.

The FFLL API has room for improvement. It's inability to read FCL files with OR statements leads to unnecessarily long sets of rules. Development of fuzzy control rules can be shortened considerably if this one constraint was removed.

6 FUTURE WORK

There are several areas that could be refined. The first is utilizing fuzzy logic to control cross-ice positioning. Currently, cross-ice movement is dictated by a simple puck-following algorithm.

Also, more players could be added to the simulation that have roles that differ from the defenseman. Rules could be written for forwards and goalies. It would be interesting to see how those roles react with the defensemen.

Finally, one could add in more finer and detailed behaviour. The current system deals with coarse positioning and general movement. It does not dictate if things like bodychecks, puck stealing, shot blocking, and fighting occur. There could be layers of fuzzy logic control added to deal with different levels of behaviour.

REFERENCES

[1] McCusky, Mason, "Fuzzy Logic for Video Games," *Game Programming Gems*, Charles River Media, 2000.

[2] Dybsand, Eric, "A Generic Fuzzy State Machine in C++," *Game Programming Gems 2*, Charles River Media, 2001.

[3] Free Fuzzy Logic Library, http://ffll.sourceforge.net/api, September 2001.

[4] The Science of Hockey: Skating, http://www.exploratorium.edu/hockey/skating2.html, September, 2004.

[5] International Electrotechnical Commission IEC 61131 Draft 1.0, http://www.fuzzytech.com/binaries/ieccd1.pdf, September 2001.

[6] Spark! Viewer, Louder Than a Bomb Software, http://www.louderthanabomb.com/spark.htm, July 2004.

APPENDIX A: FCL FILES

```
(* FCL File for Desired_Position, conservative defenseman *)

FUNCTION_BLOCK

VAR_INPUT
	Puck_Position		REAL; (* RANGE(-100 .. 100) *)
	Possession			REAL; (* RANGE(-1 .. 1) *)
	Game_Score			REAL; (* RANGE(-3 .. 3) *)
	Game_Time			REAL; (* RANGE(0 .. 60) *)
END_VAR

VAR_OUTPUT
	Desired_Position	REAL; (* RANGE(-100 .. 100) *)
END_VAR

FUZZIFY Puck_Position
	TERM DDEF := (-100, 0) (-100, 1) (-50, 0);
	TERM SDEF := (-66, 0) (-42, 1) (-18, 0);
	TERM DNEU := (-23.5, 0) (-13.5, 1) (-3.5, 0);
	TERM NEU := (-10, 0) (0, 1) (10, 0);
	TERM ONEU := (3.5, 0) (13.5, 1) (23.5, 0);
	TERM SOFF := (18, 0) (42, 1) (66, 0);
	TERM DOFF := (56, 0) (66, 1)(100, 1) (100, 0);
END_FUZZIFY

FUZZIFY Possession
	TERM OPP := -1;
	TERM NEU := 0;
	TERM OWN := 1;
END_FUZZIFY

FUZZIFY Game_Score
	TERM BEH := (-3, 0) (-3, 1) (-1, 1) (-1, 0);
	TERM TIE := 0;
	TERM AHE := (1, 0) (1, 1) (3, 1) (3, 0);
END_FUZZIFY

FUZZIFY Game_Time
	TERM 1ST := (0, 0) (0, 1) (19, 1) (20, 0);
	TERM 2ND := (20, 0) (20, 1) (39, 1) (40, 0);
	TERM 3RD := (40, 0) (40, 1) (50, 1) (60, 0);
	TERM END := (50, 0) (55, 1) (60, 1) (60, 0);
END_FUZZIFY

FUZZIFY Desired_Position
	TERM DDEF := (-100, 0) (-100, 1) (-50, 0);
	TERM SDEF := (-66, 0) (-42, 1) (-18, 0);
	TERM DNEU := (-23.5, 0) (-13.5, 1) (-3.5, 0);
	TERM NEU := (-10, 0) (0, 1) (10, 0);
	TERM ONEU := (3.5, 0) (13.5, 1) (23.5, 0);
	TERM SOFF := (18, 0) (42, 1) (66, 0);
	TERM DOFF := (56, 0) (66, 1)(100, 1) (100, 0);
END_FUZZIFY

DEFUZZIFY Desired_Position
	METHOD: COG;
END_DEFUZZIFY

RULEBLOCK first
	AND:MIN;
	ACCU:MAX;
RULE 0: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST)
THEN (Desired_Position IS SOFF);
RULE 1: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND)
THEN (Desired_Position IS SOFF);
RULE 2: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD)
THEN (Desired_Position IS SOFF);
RULE 3: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS
END) THEN (Desired_Position IS SOFF);
```

RULE 4: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 5: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 6: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 7: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 8: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 9: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 10: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 11: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 12: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 13: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 14: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 15: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 16: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 17: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 18: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 19: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 20: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 21: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 22: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 23: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 24: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 25: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 26: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 27: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 28: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 29: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 30: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 31: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 32: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 33: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 34: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 35: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 36: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 37: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 38: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);

RULE 39: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 40: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 41: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 42: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 43: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 44: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 45: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 46: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 47: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 48: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 49: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 50: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 51: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 52: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 53: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 54: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 55: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 56: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 57: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 58: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 59: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 60: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 61: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 62: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 63: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 64: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 65: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 66: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 67: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 68: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 69: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 70: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 71: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 72: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 73: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);

RULE 74: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 75: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 76: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 77: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 78: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 79: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 80: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 81: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 82: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 83: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 84: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 85: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 86: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 87: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 88: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 89: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 90: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 91: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 92: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 93: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 94: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 95: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 96: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 97: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 98: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 99: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 100: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 101: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 102: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 103: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 104: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 105: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 106: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 107: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 108: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);

RULE 109: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 110: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 111: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 112: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 113: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 114: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 115: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 116: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 117: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 118: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 119: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 120: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 121: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 122: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 123: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 124: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 125: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 126: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 127: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 128: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 129: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 130: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 131: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 132: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 133: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 134: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 135: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 136: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 137: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 138: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 139: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 140: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 141: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 142: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 143: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);

RULE 144: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 145: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 146: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 147: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 148: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 149: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 150: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 151: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 152: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 153: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 154: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 155: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 156: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 157: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 158: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 159: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 160: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 161: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 162: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 163: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 164: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 165: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 166: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 167: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 168: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 169: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 170: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 171: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 172: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 173: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 174: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 175: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 176: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 177: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 178: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);

RULE 179: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 180: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 181: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 182: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 183: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 184: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 185: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 186: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 187: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 188: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 189: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 190: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 191: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 192: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 193: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 194: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 195: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 196: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 197: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 198: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 199: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 200: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 201: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 202: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 203: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 204: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 205: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 206: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 207: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 208: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 209: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 210: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 211: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 212: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 213: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);

RULE 214: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 215: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 216: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 217: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 218: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 219: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 220: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 221: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 222: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 223: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 224: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 225: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 226: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 227: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 228: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 229: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 230: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 231: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 232: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 233: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 234: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 235: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 236: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 237: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 238: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 239: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 240: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 241: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 242: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 243: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 244: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 245: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 246: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 247: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 248: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);

```
RULE 249: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS
2ND) THEN (Desired_Position IS DDEF);
RULE 250: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS
3RD) THEN (Desired_Position IS DDEF);
RULE 251: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS
END) THEN (Desired_Position IS DDEF);
END_RULEBLOCK

END_FUNCTION_BLOCK
```

```
(* FCL File for Desired_Position, aggressive defenseman *)

FUNCTION_BLOCK

VAR_INPUT
        Puck_Position       REAL; (* RANGE(-100 .. 100) *)
        Possession                  REAL; (* RANGE(-1 .. 1) *)
        Game_Score                  REAL; (* RANGE(-3 .. 3) *)
        Game_Time                   REAL; (* RANGE(0 .. 60) *)
END_VAR

VAR_OUTPUT
        Desired_Position    REAL; (* RANGE(-100 .. 100) *)
END_VAR

FUZZIFY Puck_Position
        TERM DDEF := (-100, 0) (-100, 1) (-50, 0);
        TERM SDEF := (-66, 0) (-42, 1) (-18, 0);
        TERM DNEU := (-23.5, 0) (-13.5, 1) (-3.5, 0);
        TERM NEU := (-10, 0) (0, 1) (10, 0);
        TERM ONEU := (3.5, 0) (13.5, 1) (23.5, 0);
        TERM SOFF := (18, 0) (42, 1) (66, 0);
        TERM DOFF := (56, 0) (66, 1)(100, 1) (100, 0);
END_FUZZIFY

FUZZIFY Possession
        TERM OPP := -1;
        TERM NEU := 0;
        TERM OWN := 1;
END_FUZZIFY

FUZZIFY Game_Score
        TERM BEH := (-3, 0) (-3, 1) (-1, 1) (-1, 0);
        TERM TIE := 0;
        TERM AHE := (1, 0) (1, 1) (3, 1) (3, 0);
END_FUZZIFY

FUZZIFY Game_Time
        TERM 1ST := (0, 0) (0, 1) (19, 1) (20, 0);
        TERM 2ND := (20, 0) (20, 1) (39, 1) (40, 0);
        TERM 3RD := (40, 0) (40, 1) (50, 1) (60, 0);
        TERM END := (50, 0) (55, 1) (60, 1) (60, 0);
END_FUZZIFY

FUZZIFY Desired_Position
        TERM DDEF := (-100, 0) (-100, 1) (-50, 0);
        TERM SDEF := (-66, 0) (-42, 1) (-18, 0);
        TERM DNEU := (-23.5, 0) (-13.5, 1) (-3.5, 0);
        TERM NEU := (-10, 0) (0, 1) (10, 0);
        TERM ONEU := (3.5, 0) (13.5, 1) (23.5, 0);
        TERM SOFF := (18, 0) (42, 1) (66, 0);
        TERM DOFF := (56, 0) (66, 1)(100, 1) (100, 0);
END_FUZZIFY

DEFUZZIFY Desired_Position
        METHOD: COG;
END_DEFUZZIFY

RULEBLOCK first
        AND:MIN;
        ACCU:MAX;
RULE 0: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST)
THEN (Desired_Position IS SOFF);
RULE 1: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND)
THEN (Desired_Position IS SOFF);
RULE 2: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD)
THEN (Desired_Position IS SOFF);
RULE 3: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS
END) THEN (Desired_Position IS SOFF);
```

RULE 4: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 5: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 6: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 7: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 8: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 9: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 10: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 11: IF (Puck_Position IS DOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 12: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 13: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 14: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 15: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 16: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 17: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 18: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 19: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 20: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 21: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 22: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 23: IF (Puck_Position IS DOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 24: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 25: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 26: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 27: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 28: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 29: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 30: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 31: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 32: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SOFF);
RULE 33: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SOFF);
RULE 34: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SOFF);
RULE 35: IF (Puck_Position IS DOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 36: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 37: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 38: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);

RULE 39: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 40: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 41: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 42: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 43: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 44: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 45: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 46: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 47: IF (Puck_Position IS SOFF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 48: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 49: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 50: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 51: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 52: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 53: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 54: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 55: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 56: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 57: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 58: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 59: IF (Puck_Position IS SOFF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 60: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 61: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 62: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 63: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SOFF);
RULE 64: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 65: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 66: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 67: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS ONEU);
RULE 68: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS ONEU);
RULE 69: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS ONEU);
RULE 70: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS ONEU);
RULE 71: IF (Puck_Position IS SOFF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 72: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 73: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);

RULE 74: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 75: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 76: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 77: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 78: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 79: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 80: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 81: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 82: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 83: IF (Puck_Position IS ONEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 84: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 85: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 86: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 87: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 88: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 89: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 90: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 91: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 92: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 93: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 94: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 95: IF (Puck_Position IS ONEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 96: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 97: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 98: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 99: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 100: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 101: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 102: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 103: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS NEU);
RULE 104: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS NEU);
RULE 105: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS NEU);
RULE 106: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS NEU);
RULE 107: IF (Puck_Position IS ONEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 108: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);

RULE 109: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 110: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 111: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 112: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 113: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 114: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 115: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 116: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 117: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 118: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 119: IF (Puck_Position IS NEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 120: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 121: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 122: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 123: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 124: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 125: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 126: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 127: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 128: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 129: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 130: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 131: IF (Puck_Position IS NEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 132: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 133: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 134: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 135: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 136: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 137: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 138: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 139: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 140: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 141: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 142: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 143: IF (Puck_Position IS NEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);

RULE 144: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 145: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 146: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 147: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 148: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 149: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 150: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 151: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DNEU);
RULE 152: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DNEU);
RULE 153: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DNEU);
RULE 154: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DNEU);
RULE 155: IF (Puck_Position IS DNEU) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 156: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 157: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 158: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 159: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 160: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 161: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 162: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 163: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 164: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 165: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 166: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 167: IF (Puck_Position IS DNEU) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 168: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 169: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 170: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 171: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 172: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 173: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 174: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 175: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 176: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 177: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 178: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);

RULE 179: IF (Puck_Position IS DNEU) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 180: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 181: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 182: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 183: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 184: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 185: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 186: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 187: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 188: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 189: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 190: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 191: IF (Puck_Position IS SDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 192: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 193: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 194: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 195: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 196: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 197: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 198: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 199: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 200: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 201: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 202: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 203: IF (Puck_Position IS SDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 204: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 205: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 206: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 207: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 208: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 209: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);
RULE 210: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 211: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 212: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS SDEF);
RULE 213: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS SDEF);

RULE 214: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS SDEF);
RULE 215: IF (Puck_Position IS SDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS SDEF);
RULE 216: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 217: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 218: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 219: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 220: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 221: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 222: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 223: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 224: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 225: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 226: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 227: IF (Puck_Position IS DDEF) AND (Possession IS OWN) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 228: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 229: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 230: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 231: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 232: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 233: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 234: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 235: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 236: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 237: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 238: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 239: IF (Puck_Position IS DDEF) AND (Possession IS NEU) AND (Game_Score IS AHE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 240: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 241: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 242: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 243: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS BEH) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 244: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);
RULE 245: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 2ND) THEN (Desired_Position IS DDEF);
RULE 246: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS 3RD) THEN (Desired_Position IS DDEF);
RULE 247: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS TIE) AND (Game_Time IS END) THEN (Desired_Position IS DDEF);
RULE 248: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS 1ST) THEN (Desired_Position IS DDEF);

```
RULE 249: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS
2ND) THEN (Desired_Position IS DDEF);
RULE 250: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS
3RD) THEN (Desired_Position IS DDEF);
RULE 251: IF (Puck_Position IS DDEF) AND (Possession IS OPP) AND (Game_Score IS AHE) AND (Game_Time IS
END) THEN (Desired_Position IS DDEF);
END_RULEBLOCK

END_FUNCTION_BLOCK
```

```
(* FCL File for Velocity *)

FUNCTION_BLOCK

VAR_INPUT
        Separation                      REAL; (* RANGE(-200 .. 200) *)
END_VAR

VAR_OUTPUT
        Velocity            REAL; (* RANGE(-29.3 .. 29.3) *)
END_VAR

FUZZIFY Separation
        TERM LG_NEG := (-200, 0) (-200, 1) (-110, 1) (-40, 0);
        TERM MD_NEG := (-130,0) (-110, 1) (-58, 1) (-15, 0);
        TERM SM_NEG := (-50, 0) (-10, 1) (0, 0);
        TERM ZERO := (-15, 0) (0, 1) (15, 0);
        TERM SM_POS := (0, 0) (10, 1) (50, 0);
        TERM MD_POS := (15, 0) (58, 1) (110, 1) (130, 0);
        TERM LG_POS := (40, 0) (110, 1)(200, 1) (200, 0);
END_FUZZIFY

FUZZIFY Velocity
        TERM LG_NEG := (-29.3, 0) (-29.3, 1) (-26, 0);
        TERM MD_NEG := (-26, 0) (-15, 1) (-10, 0);
        TERM SM_NEG := (-13, 0) (-8, 1) (-1, 0);
        TERM ZERO := (-2, 0) (0, 1) (2, 0);
        TERM SM_POS := (1, 0) (8, 1) (13, 0);
        TERM MD_POS := (10, 0) (15, 1) (26, 0);
        TERM LG_POS := (26, 0) (29.3, 1) (29.3, 0);
END_FUZZIFY

DEFUZZIFY Velocity
        METHOD: COG;
END_DEFUZZIFY

RULEBLOCK first
        AND:MIN;
        ACCU:MAX;
        RULE 0: IF (Separation IS LG_NEG) THEN (Velocity IS LG_NEG);
        RULE 1: IF (Separation IS MD_NEG) THEN (Velocity IS LG_NEG);
        RULE 2: IF (Separation IS SM_NEG) THEN (Velocity IS SM_NEG);
        RULE 3: IF (Separation IS ZERO) THEN (Velocity IS ZERO);
        RULE 4: IF (Separation IS SM_POS) THEN (Velocity IS SM_POS);
        RULE 5: IF (Separation IS MD_POS) THEN (Velocity IS LG_POS);
        RULE 6: IF (Separation IS LG_POS) THEN (Velocity IS LG_POS);
END_RULEBLOCK

END_FUNCTION_BLOCK
```

APPENDIX B : LIST OF SIMULATION COMMANDS

Changing object positions

The user can change the position of the defensemen, teammate, opponent, and puck by clicking on and dragging the particular simulation object.

Moving with the puck – "g" key

The simulation allows the user to move both the teammate and the puck or the opponent and the puck at the same time. This effectively creates the effect of a player skating with puck. To do this, hit the "g" key with the simulation window active. This will enable group selection. Now click and drag either the teammate (in green) or the opponent in (yellow) to the puck. Once contact is made, the puck will stay with the object. To disable group selection, hit the "g" key again.

Changing the game score – "s" key

With the simulation window active, hit the "s" key to change the score of the game. Then bring up the console window and answer the questions to change the score.

Changing the game time – "t" key

With the simulation window active, hit the "t" key to change the game time. Then bring up the console window and answer the questions to change the game time.

Quitting the simulation – "q" key

To quit the simulation at any time, hit the "q" key.