



**UNIVERSIDADE FEDERAL DE ITAJUBÁ**  
Programa de Pós-Graduação em Engenharia Elétrica

**AJUSTE DE REGRAS E FUNÇÕES  
DE PERTINÊNCIA UTILIZANDO  
ALGORITMOS GENÉTICOS**

Dissertação apresentada à  
Universidade Federal de Itajubá,  
para obtenção do título de Mestre  
em Ciências em Engenharia  
Elétrica.

**Patrícia Cerávolo Rodrigues de Paiva Nunes Oliveira**

Itajubá

2004

**Patrícia Cerávolo Rodrigues de Paiva Nunes Oliveira**

**AJUSTE DE REGRAS E FUNÇÕES  
DE PERTINÊNCIA UTILIZANDO  
ALGORITMOS GENÉTICOS**

Dissertação apresentada à  
Universidade Federal de Itajubá,  
para obtenção do título de Mestre  
em Ciências em Engenharia  
Elétrica.

**Área de concentração:**  
Automação e Sistemas Elétricos

**Orientador:**  
Germano Lambert Torres

**Co-orientador:**  
Alexandre Rasi Aoki

Itajubá

Julho/2004

***Ao meu querido filho Elias,  
razão deste trabalho***

## Agradecimentos

A Deus pelo dom da vida, pela saúde e por tudo de bom que temos.

Ao meu marido Emerson, pelo apoio, companheirismo, amizade e amor.

Ao meu filhinho Elias, por existir e ser alegria em minha vida.

Aos meus pais, Víctor e Regina, pelo amor incondicional, ajuda em todos os momentos de minha vida, base de amor que sei que sempre poderei contar.

Aos meus queridos Juliana e Alex, pela amizade e pela disposição em me ajudar em tudo.

Ao professor Germano Lambert Torres pela dedicação, orientação e suporte em todos os momentos deste trabalho.

Ao amigo Alexandre Rasi Aoki e à Milena pelo incentivo e amizade que resultaram na realização deste trabalho.

A todos, que direta ou indiretamente, contribuíram com seu apoio e motivação para a concretização deste sonho.

## Sumário

<b>Agradecimentos .....</b>	<b>iv</b>
<b>Resumo .....</b>	<b>vii</b>
<b>Abstract .....</b>	<b>viii</b>
<b>Lista de Figuras .....</b>	<b>ix</b>
<b>Lista de Tabelas .....</b>	<b>xi</b>
<b>Capítulo 1 – Introdução .....</b>	<b>1</b>
<b>Capítulo 2 – Teoria sobre Algoritmos Genéticos.....</b>	<b>3</b>
2.1. Introdução.....	3
2.2. Implementação dos Algoritmos Evolucionários .....	4
2.3. Algoritmos Genéticos.....	6
2.4. Características dos Operadores Genéticos .....	9
2.4.1. Representação das Variáveis.....	9
2.4.2. Inicialização da População.....	10
2.4.3. Cálculo da Função de Avaliação.....	10
2.4.4. Seleção através da Roleta da Sorte.....	11
2.4.5. Cruzamento .....	12
2.4.6. Mutação.....	13
2.4.7. Tamanho da População .....	14
<b>Capítulo 3 – Descrição da Seqüência de Execução do Trabalho.....</b>	<b>15</b>
3.1. Base do Trabalho.....	15
3.2. Regras.....	18
3.2.1. Componentes dos Algoritmos Genéticos.....	18
3.2.2. Representação das Variáveis.....	19
3.2.3. Inicialização da População.....	20
3.2.4. Função de Avaliação .....	21
3.2.5. Cruzamento .....	21
3.2.6. Mutação.....	22
3.2.7. Seleção através da Roleta da Sorte.....	23
3.2.8. Tamanho da População .....	24
3.2.9. Critério de Parada.....	24
3.3. O Cromossomo.....	25

3.4. Regras juntamente com as Funções de Pertinência (FP's).....	25
3.5. Testes.....	26
3.6. Roleta Forçada.....	27
<b>Capítulo 4 – Descrição do Módulo Treinamento Genético.....</b>	<b>28</b>
4.1. Apresentação das Funcionalidades.....	28
4.1.1. A Opção Definir Posições Iniciais .....	29
4.1.2. A Opção Treinamento - Regras.....	29
4.2. Apresentação do Algoritmo.....	32
<b>Capítulo 5 – Testes.....</b>	<b>44</b>
5.1. Introdução.....	44
5.2. O controle GEN1 .....	45
5.2.1. Treinamento de Regras para o controle GEN1 .....	46
5.2.2. Treinamento de Regras + FP's para o controle GEN1.....	48
5.3. O controle GEN2.....	51
5.3.1. Treinamento de Regras para o controle GEN2 .....	53
5.3.2. Treinamento de Regras + FP's para o controle GEN2.....	55
5.4. Resultados de Simulações .....	57
<b>Capítulo 6 – Conclusão e Trabalhos Futuros.....</b>	<b>60</b>
<b>Referências Bibliográficas .....</b>	<b>62</b>
<b>Anexos .....</b>	<b>64</b>

## Resumo

O propósito deste trabalho é apresentar uma estratégia para o ajuste automático das regras e das funções de pertinência usando Algoritmos Genéticos. Para tal, foi desenvolvido e implementado um algoritmo que transforma as regras e as funções de pertinências em cromossomos que são submetidos a uma evolução, cruzamento e mutação. A idéia geral é obter uma nova família de regras e funções de pertinência que possa melhor controlar um processo, otimizando o resultado final.

O algoritmo proposto foi incorporado ao Pacote Computacional para o Ensino da Lógica Difusa anteriormente desenvolvido que tem o objetivo de ensinar a lógica difusa para estudantes de controle. Este pacote contém todas as instruções necessárias para os usuários compreenderem os princípios do controle difuso. O principal objetivo do pacote é estacionar um veículo, partindo de qualquer posição inicial, em uma garagem. Para cumprir esta tarefa, os estudantes devem inicialmente desenvolver um conjunto de regras de controle difuso e funções de pertinência que definirão a trajetória do veículo. Os processos de fuzzificação e defuzzificação das variáveis são realizados pelo programa sem a interferência do usuário.

A escolha do método de otimização Algoritmos Genéticos é devido à melhora que esses algoritmos trazem ao desempenho de um controlador difuso, comprovado pelos resultados obtidos neste trabalho. Esses algoritmos vêm sendo aplicados com sucesso nos mais diversos problemas de otimização e aprendizado de máquina.

## **Abstract**

The purpose of this paper is to present a strategy for an automatic rules fitting and an automatic membership function fitting using genetic algorithms. For that, an algorithm has been developed and implemented to transform the rules and the fuzzy membership function in chromosomes, which are submitted to an evolution, crossover and mutation. The main idea is to fit a new family of rules and membership functions, which can establish a better system control, optimizing the final result.

The proposed algorithm has been incorporated to the existent working Computational Package for Teaching Fuzzy Control, with the objective of teaching Fuzzy Logic to the control students. The package contains all required instructions for the users to gain an understanding of fuzzy control principles. The main objective of the package is to park a car, approaching from any direction, in a parking lot. To accomplish this task the students must first develop sets of fuzzy-control rules and membership functions which define the trajectory of the car. Processes, such as fuzzification and defuzzification of the variables, are performed by the program without the interference of the user.

The choice of the optimization method genetic algorithms is because of the better they bring to the performance of a fuzzy controller, proved for the obtained results of this work. These algorithms have been used with success in many varied problems of optimization and machine learning.

## Lista de Figuras

2.1 – Cruzamento de 1 ponto antes e depois do cruzamento.....	12
3.1 – Tela principal do programa .....	15
3.2 – Conjunto de funções de pertinência .....	16
3.3 – Parte das regras difusas editadas pelo usuário.....	16
3.4 – Exemplo de ajuste de uma função de pertinência .....	17
3.5 – Janela de edição de regras .....	19
3.6 – Criação de um novo cromossomo a partir do cromossomo da regra original	20
3.7 – Cruzamento efetuado nas regras.....	22
3.8 – Exemplo de uma mutação ocorrida no cromossomo das regras.....	22
3.9 – Representação da roleta.....	24
4.1 – Menu treinamento genético .....	28
4.2 – Posições iniciais para treinamento genético .....	29
4.3 – Parâmetros genéticos .....	30
4.4 – Melhores resultados.....	31
4.5 – Simulação sem treinamento genético .....	34
4.6 – Funções de pertinência sem ajuste .....	35
4.7 – Um retrato das regras sem ajuste.....	35
4.8 – Melhores resultados do treinamento genético .....	41
4.9 – Simulação após treinamento genético. ....	42
4.10 – Um retrato das regras após o ajuste.....	42
4.11 – Funções de pertinência após o ajuste.....	43
5.1 – Funções de pertinência do controle RE.....	44
5.2 – Posições iniciais de treinamento do controle GEN1 .....	45
5.3 – Simulações sem treinamento .....	46
5.4 – Comparações das trajetórias do treinamento de regras de GEN1 .....	47
5.5 – Comparações das trajetórias do treinamento de regras + FP's de GEN1 .....	49
5.6 – Funções de pertinência originais .....	50
5.7 – Funções de pertinência após o ajuste de FP's .....	50
5.8 – Posições iniciais do treinamento de GEN2 .....	51
5.9 – Simulações sem treinamento .....	52
5.10 – Comparações das trajetórias do treinamento de regras de GEN2. ....	54

5.11 – Comparações das trajetórias do treinamento de regras + FP's de GEN2.....	56
5.12 – Uma comparação das FP's originais com as FP's após o ajuste .....	57

## Lista de Tabelas

2.1 – População inicializada aleatoriamente .....	7
2.2 – Avaliação da população .....	7
2.3 – Seleção dos indivíduos .....	8
2.4 – Nova população formada .....	8
2.5 – Cruzamento e mutação na nova população .....	8
2.6 – Avaliação da nova população .....	9
3.1 – Critério de seleção .....	23
4.1 – Primeira geração .....	37
4.2 – Segunda geração .....	38
4.3 – Terceira geração .....	40
4.4 – Última geração .....	41
5.1 – Posições iniciais para o controle GEN1 .....	45
5.2 – Parâmetros genéticos para o treinamento de GEN1 .....	46
5.3 – Iterações após o treinamento genético de GEN1 .....	47
5.4 – Parâmetros genéticos para o treinamento de GEN1 .....	48
5.5 – Iterações após o treinamento genético de GEN1 .....	48
5.6 – Posições iniciais para o controle GEN2 .....	51
5.7 – Parâmetros genéticos para o controle GEN2 .....	53
5.8 – Iterações após o treinamento genético do controle GEN2 .....	53
5.9 – Parâmetros genéticos para o controle GEN2 .....	55
5.10 – Iterações após o treinamento genético do controle GEN2 .....	55
5.11 – Resultados de simulações .....	58

# Capítulo 1

## Introdução

A Natureza é sábia, está em constante evolução, e evolução significa mudança, modificação ao longo do tempo em busca da perfeição, buscando melhorar em pontos mais fracos, em busca do perfeito, do ótimo. Porque não utilizar-se das leis da natureza nos estudos da computação para que se possa obter melhores resultados, tal qual na Natureza onde tudo funciona tão perfeitamente.

Algoritmos Genéticos pertencem a uma das áreas da Computação Evolucionária onde o processo de otimização é adaptativo. As populações das estruturas computacionais evoluem de tal forma que o desempenho geral da população melhora na média, ou seja, a próxima geração é mais evoluída que a anterior. São poderosas técnicas de otimização e de busca estocástica aplicada a diversos tipos de problemas inspirados em mecanismos de evolução natural e genética. Isto é feito através de processos iterativos, onde cada iteração é chamada de geração. A cada geração temos uma população de indivíduos (possíveis soluções para o problema) que trocam informações entre si e, ocasionalmente, sofrem mutações. A mutação é a chave para a evolução. Após várias gerações, a evolução dá um salto rumo a indivíduos mais aptos a se adaptarem ao meio. Esses indivíduos são avaliados e melhorados de geração a geração.

Neste trabalho foi melhorado o Módulo de Treinamento Genético do Pacote Computacional para Ensino da Lógica Difusa [1]. Os algoritmos genéticos são utilizados para encontrar as melhores regras juntamente com as melhores funções de pertinência que fazem com que o veículo estacione em uma garagem no menor tempo possível. As regras e as funções de pertinência são os indivíduos da população (cromossomos), avaliados de acordo com o número de iterações necessárias para o veículo estacionar na garagem, número este que deve ser o menor possível.

O Pacote Computacional para o Ensino da Lógica Difusa é um software desenvolvido com o objetivo de ensinar a lógica difusa para estudantes de controle [2]. Contém todas as instruções necessárias para os usuários compreenderem os princípios do controle difuso [3]. O principal objetivo do pacote é estacionar um veículo, partindo de qualquer posição inicial, em uma garagem. Para cumprir esta tarefa, os estudantes devem inicialmente desenvolver um conjunto de regras de controle difuso e funções de pertinência que definirão a trajetória do veículo. Os processos de fuzzificação e defuzzificação são realizados pelo programa sem a interferência do usuário [4].

Para que se possa ter uma visão geral de como este trabalho foi conduzido, é apresentado a seguir um resumo do que cada capítulo apresenta. O Capítulo 2 relata a teoria sobre os Algoritmos Genéticos, como implementá-los e as características dos parâmetros que influenciam no comportamento de tais algoritmos.

O Capítulo 3 descreve a seqüência de como foi executado este trabalho, quais foram as bases para a realização deste, como foi a aplicação dos algoritmos genéticos ao problema proposto, as fases de transformação do programa e um breve resumo dos testes feitos com o objetivo de otimizar cada vez mais o treinamento genético.

O Capítulo 4 descreve o módulo de treinamento genético apresentando suas funcionalidades juntamente com as telas de interface com o usuário. Mostra também o algoritmo passo a passo exemplificando uma simulação com as telas de antes e depois do treinamento para que se possa entendê-lo melhor.

O Capítulo 5 ilustra os testes realizados com os controles difusos que foram ajustados pelos algoritmos genéticos para uma avaliação objetiva dos resultados.

O Capítulo 6 apresenta as conclusões e trabalhos futuros que possam dar continuidade aos resultados deste trabalho.

## Capítulo 2

### Teoria sobre Algoritmos Genéticos

#### 2.1. Introdução

Existem vários componentes em toda tarefa de busca e otimização, entre eles: o espaço de busca [5]. Nele são consideradas todas as possibilidades de solução de um determinado problema, incluindo a função de avaliação, ou seja, uma maneira de avaliar os membros do espaço de busca.

Dentre os métodos de busca e otimização existentes, a técnica tradicional inicia-se com um único candidato que, iterativamente, é manipulado utilizando algumas heurísticas (regras estáticas) diretamente associadas ao problema a ser solucionado. Geralmente, estes processos heurísticos não são algorítmicos e sua simulação em computadores pode ser muito complexa pelo fato destes métodos não serem suficientemente robustos.

Neste trabalho foram utilizadas as técnicas de computação evolucionária, que diferem dos paradigmas de otimização e procura tradicionais de três maneiras [6]:

*a) Utilizam população de pontos (que são as soluções em potencial) em sua procura.*

A maioria dos paradigmas de otimização tradicionais move de um ponto a outro no hiperespaço de decisão usando alguma regra determinística. Uma das desvantagens disto é a probabilidade de chegada em um ótimo local.

Os paradigmas da computação evolucionária utilizam uma população de pontos (vetores do hiperespaço) que geram uma nova população com o mesmo número de membros a cada época (geração), ou seja, operam sobre uma população de candidatos em paralelo. Assim, elas podem fazer a busca em diferentes áreas do espaço de solução, alocando um número de membros apropriado para a busca em várias regiões. Com isso, muitos máximos ou mínimos podem ser explorados simultaneamente, diminuindo a probabilidade de estagnação do processo. Os operadores tais como o cruzamento e a mutação aumentam efetivamente esta capacidade de busca paralela permitindo à busca “pular” diretamente de um hiperespaço promissor para outro.

*b) Usam a informação direta da função de avaliação ao invés de funções derivativas.*

Muitos paradigmas de busca “subida de encosta” requerem o cálculo de derivadas para a exploração de máximos locais.

Os paradigmas da computação evolucionária não requerem informação auxiliar ao problema em questão, tais como funções derivativas. A função de avaliação de cada membro da população é calculada a partir do valor da função a ser otimizada, e é comum usar a saída da função como a medida da função de avaliação. A função de avaliação é a medida direta do desempenho do membro individual da população na função a ser otimizada.

*c) Usam regras de transição probabilísticas ao invés de determinísticas.*

O fato dos paradigmas da computação evolucionária usarem regras de transição probabilística certamente não significa que uma busca aleatória rigorosa está sendo realizada. Melhor dizendo, os operadores estocásticos são apresentados às operações que direcionam a busca através de regiões do hiperespaço prováveis de terem altos valores da função de avaliação. Por exemplo, a reprodução (seleção) é freqüentemente realizada com a probabilidade proporcional ao valor da função de avaliação do indivíduo.

Além disso, essas técnicas codificam os parâmetros (em binários ou outros símbolos), ao invés de trabalhar com os parâmetros em si.

## **2.2. Implementação dos Algoritmos Evolucionários**

As ferramentas da computação evolucionária freqüentemente seguem um procedimento similar:

*a) Inicializam uma população.*

A inicialização é comumente feita com valores aleatórios. No caso de os parâmetros serem representados por *strings* (palavras) binários, seria simplesmente gerar *strings* aleatórias de 0's e 1's (com uma probabilidade uniforme para cada um) e com um tamanho fixo. Entretanto, existem aplicações onde é mais conveniente o uso de representações por inteiros como apresentado mais adiante neste trabalho. Algumas vezes é praticável também semear a população com valores “talentosos”, que parecem prometer algo, ou seja, que com relação à região do hiperespaço se encontram perto do ótimo. O número total de indivíduos escolhido para compor a população se encontra na faixa de algumas dúzias ou algumas centenas.

*b) Calculam a função de avaliação para cada indivíduo na população.*

A função de avaliação é proporcional ao valor de saída da função a ser otimizada (podendo também ser derivado da combinação de várias funções de saída). A função de avaliação toma como entradas as saídas de uma ou mais funções e sua saída é uma probabilidade de reprodução. A maior parte do tempo computacional usada é para a validação da função de avaliação.

*c) Selecionam os indivíduos para formar uma nova população.*

A seleção dos indivíduos para a reprodução (para constituir uma nova população) é baseada nos valores da função de avaliação. Quanto mais alto for o valor, mais “selecionável” será o indivíduo.

*d) Fazem operações evolucionárias, tais como cruzamento e mutação na população.*

Um conjunto de operações é necessário para que, dada uma população, se consiga gerar populações sucessivas que (espera-se) melhorem sua aptidão com o tempo. Estes operadores são: cruzamento (crossover) e mutação. Eles são utilizados para assegurar que a nova geração seja totalmente nova, mas possui, de alguma forma, características de seus pais, ou seja, a população se diversifica e mantém características de adaptação adquiridas pelas gerações anteriores.

*e) Voltam ao passo b) até que alguma condição seja alcançada.*

O fim do algoritmo acontece quando um membro da população atinge uma função de avaliação especificada ou o algoritmo já rodou o número de vezes pré-determinado.

### **2.3. Algoritmos Genéticos**

Algoritmos Genéticos pertencem a uma das áreas da computação evolucionária, portanto sua implementação é feita de acordo com o descrito acima em 2.2. A série de operações seguidas na implementação de um paradigma de Algoritmos Genéticos é a seguinte [7]:

- 1) Inicialização da população.
- 2) Cálculo da função avaliação para cada indivíduo da população.
- 3) Seleção dos indivíduos para a formação de uma nova população.
- 4) Realização de cruzamento e mutação na população.
- 5) Volta ao passo 2) até que alguma condição seja encontrada.

São algoritmos de busca que refletem de maneira primitiva alguns dos processos de evolução natural. Frequentemente, proporcionam mecanismos de procura efetivos que podem ser usados em aplicações de otimização ou classificação.

Como observado no item 2.1, os paradigmas da computação evolucionária trabalham com uma população de pontos, ao invés de um único ponto; cada ponto é um vetor no hiperespaço representando uma potencial (ou candidata) solução para o problema de otimização ou classificação. Uma população é um conjunto de vetores no hiperespaço. Cada vetor é um indivíduo na população, chamado de cromossomo devido à analogia com a evolução genética dos organismos.

Os Algoritmos Genéticos operam pela validação de uma população de palavras binárias e os sobreviventes selecionados pela sua função de avaliação serão os membros da população mais prováveis de sobreviver. Os sobreviventes serão cruzados e alguma mutação será feita nos cromossomos. A recombinação de material genético é um poderoso método para a adaptação.

A seguir é apresentado um resumo da teoria dos Algoritmos Genéticos através de um exemplo:

1) Inicialização da população.

É gerada uma população formada por um conjunto aleatório de indivíduos que podem ser vistos como possíveis soluções do problema. Neste exemplo, seja uma população de 4 indivíduos conforme a Tabela 2.1, cujo problema seja maximizar a função  $f(x)=x^2$ , onde  $x \in X = \{0 1 2 \dots 63\}$ .

<i>I</i>	Indivíduo $x_i$
1	001101
2	011000
3	001000
4	010011

Tabela 2.1 População inicializada aleatoriamente

2) Cálculo da função avaliação para cada indivíduo da população.

Tal função tem o papel de avaliar o nível de aptidão (adaptação) de cada cromossomo. No exemplo, o cromossomo de maior aptidão é o  $2^0$  ( $f(x)=576$ ), ilustrado pela Tabela 2.2.

<i>I</i>	Indivíduo $x_i$	$x$	$f(x)$
1	001101	13	169
2	011000	24	576
3	001000	8	64
4	010011	19	361
Total			1170

Tabela 2.2 Avaliação da população

3) Seleção dos indivíduos para a formação de uma nova população.

Durante o processo evolutivo, esta população é avaliada: para cada indivíduo é dado um índice refletindo sua habilidade de adaptação a determinado ambiente. A Tabela 2.3 ilustra a seleção dos indivíduos através da roleta da sorte. Uma porcentagem dos mais adaptados é mantida (mostrado na Tabela 2.4), enquanto os outros são descartados.

Indivíduo $x_i$	$x$	$f(x)$	Aptidão Relativa	Descendentes Esperados	Seleção		
					Reprodução	Roleta	
1	001101	13	169	0,14	0,56	0	0
2	011000	24	576	0,49	1,96	1	0
3	001000	8	64	0,06	0,24	0	1
4	010011	19	361	0,31	1,24	1	1
Total			1170	1,00	4,00	2	2

Tabela 2.3 Seleção dos indivíduos

$I$	Indivíduo $x_i$	$x$
2	011000	24
3	001000	8
4	010011	19
4	010011	19

Tabela 2.4 Nova população formada

4) Realização de cruzamento e mutação na população.

Os membros mantidos pela seleção sofrem modificações em suas características fundamentais através de mutações e cruzamento, gerando descendentes para a próxima geração, conforme a Tabela 2.5.

Indivíduo $x_i$	Emparelhamento	Ponto de Crossover	Nova População	Mutação	Nova População	
2	011000	3	2	010011	-	010011
3	001000	2	4	001011	2	011011
4	010011	1	4	010000	3	011000
4	010011	0	2	011000	-	011000
Total						

Tabela 2.5 Cruzamento e mutação na nova população

5) Volta ao passo 2) até que alguma condição seja encontrada.

A nova população é avaliada e o processo é repetido até que uma solução satisfatória seja encontrada. A Tabela 2.6 ilustra a avaliação da nova população formada.

<i>I</i>	Indivíduo $x_i$	Emparelhamento	Ponto de Crossover	Mutação	Nova População	$x$	$f(x)$	
2	011000	3	2	-	010011	19	361	
3	001000	2	4	2	011011	27	709	
4	010011	1	4	3	011000	24	576	
4	010011	0	2	-	011000	24	576	
Total								1646

Tabela 2.6 Avaliação da nova população

## 2.4. Características dos Operadores Genéticos

É importante também, analisar de que maneira alguns parâmetros influem no comportamento dos Algoritmos Genéticos, para que se possa estabelecê-los conforme as necessidades do problema e dos recursos disponíveis [8].

### 2.4.1 Representação das Variáveis

Dentre os tipos de representação das variáveis, também chamados de cromossomos, tem-se:

- 1) Representação Binária: representação de 0's e 1's de acordo com a faixa em que se encontra o valor que quer-se representar. Por exemplo, o valor 127 pode ser representado por um número binário de 8 bits, 01111111. Os *strings* binários são quase sempre de tamanho fixo, ou seja, não variam durante a execução. O tamanho do *string* (número de bits de um *string* binário) depende da precisão requerida para o problema a ser resolvido e de sua máxima faixa.

- 2) Representação por inteiros: é o caso dos genes dos cromossomos serem representados por números inteiros, e não cada gene ser o 0 ou o 1 pertencente ao número inteiro. Por exemplo, o cromossomo usado neste trabalho é um conjunto de genes de números inteiros, que representam os conjuntos difusos do ângulo da roda que são valores reais.

### **2.4.2 Inicialização da População**

A inicialização da população é geralmente feita estocasticamente, através de sorteio, embora seja apropriado algumas vezes inicializá-la com um ou mais indivíduos selecionados heurísticamente, metodicamente. É comum semear a população com alguns poucos membros selecionados heurísticamente e completá-la com membros escolhidos aleatoriamente. Não importa qual processo seja usado, o que importa é que a população deve representar um vasto sortimento dos indivíduos.

### **2.4.3 Cálculo da Função de Avaliação**

O cálculo dos valores da função de avaliação é conceitualmente simples, embora possa ser muito complexo implementá-lo de uma maneira que otimize a eficiência da busca no espaço problema.

Uma desvantagem no cálculo da função de avaliação é que depois que o algoritmo rodou um determinado número de gerações (algumas dúzias de vezes), as notas de aptidão dos indivíduos são bem altas. Isto diminui as diferenças das notas de aptidão entre os indivíduos que abastecem a roleta da sorte para a seleção, onde altos valores da função de avaliação terão alta probabilidade de reprodução.

Uma maneira de resolver este problema seria espaçar igualmente os valores da função de avaliação. Por exemplo, os valores da função de avaliação usados para a reprodução poderiam ser igualmente espaçados de 0 até 1, fixando o valor 1 para o melhor membro da população (com a nota mais alta), 0.875 para o segundo e 0.125 para o de nota mais baixa.

Neste caso, os membros da população estariam em um *ranking* com base na função de avaliação, e então suas classificações seriam divididas pelo número de indivíduos para proporcionar um limiar de probabilidades para a seleção. Então a função resultado não seria igual à função de avaliação, e ao invés de avaliar o desempenho real do Algoritmo Genético, a função resultado seria monitorada tão bem quanto a função de avaliação espaçada.

Um outro jeito de resolver o problema seria usar o chamado escalonamento. O escalonamento leva em conta a história recente da população e determina os valores da função de avaliação com base na comparação do desempenho dos indivíduos com os valores de desempenho médios recentes da população. Se a otimização dos Algoritmos Genéticos envolve a maximização de alguma função, então o escalonamento mantém um registro dos valores mínimos da função de avaliação obtidos das últimas gerações, os quais são utilizados como base para os valores da função de avaliação ao invés do valor 0 (que é o mínimo possível). Então, os valores da função de avaliação seriam determinados baseados na distância real relativa ao novo valor base.

#### **2.4.4 Seleção através da Roleta da Sorte**

Em Algoritmos Genéticos, o número esperado de vezes que cada indivíduo na população atual é selecionado para a nova população é proporcional à função de avaliação deste indivíduo relativo à função de avaliação média da população inteira.

Este conceito é semelhante a uma Roda da Sorte, cuja superfície é subdividida em fatias que representam a probabilidade de um indivíduo ser selecionado (o tamanho da fatia), a qual é proporcional à função de avaliação deste indivíduo. Indivíduos com altas notas de aptidão estarão mais aptos de serem selecionados (pois terão fatias maiores que os outros). Este algoritmo de seleção será repetido até que o número desejado de indivíduos tenha sido selecionado.

É possível que os indivíduos com altas notas de aptidão de uma determinada geração não sobrevivam à reprodução, cruzamento e mutação. Nesses casos, é útil usar a *estratégia elitista*, a qual assegura que esse indivíduo com uma alta nota de aptidão seja sempre copiado para a próxima geração.

## 2.4.5 Cruzamento

O mais importante operador genético utilizado é o cruzamento, tanto que suas taxas são ajustadas para valores bem mais altos que os da mutação. Outros algoritmos evolucionários usam a Mutação como método primário, mas com os Algoritmos Genéticos não é assim.

O cruzamento é baseado na metáfora da reprodução sexual, um dos maiores e mais fundamentais métodos de busca que a vida biológica possui. É o operador responsável pela recombinação de características dos pais durante a reprodução, permitindo que as próximas gerações herdem essas características. A reprodução sexual capitaliza as diferenças e semelhanças entre os indivíduos dentro das espécies, onde um indivíduo pode descender de uma linha que contém uma boa solução para um conjunto de fatores ambientais, e outro lidar melhor com outro aspecto de sobrevivência. Por exemplo, um coelho tem uma coloração branca que o protege na neve enquanto outro possui um comportamento de ficar imobilizado na presença de predadores. Juntando-se as duas características em um só coelho, a seleção natural decidirá qual dos coelhos se adaptará melhor ao ambiente, e o melhor irá sobreviver.

Existem várias maneiras de implementar o cruzamento variando seus dois atributos principais:

- 1) Tipo de cruzamento a ser implementado: o tipo mais básico existente é o cruzamento de 1 ponto, o qual envolve selecionar um ponto aleatoriamente e trocar as porções das palavras (*strings*) à direita do ponto de cruzamento. A Figura 2.1 ilustra o cruzamento de 1 ponto; o ponto de cruzamento é a linha vertical, e as porções a serem trocadas estão em negrito.

10110  <b>010</b>	10110 <b>100</b>
01001  <b>100</b>	01001 <b>010</b>
(a)	(b)

Figura 2.1 Cruzamento de 1 ponto

(a) antes do cruzamento e (b) depois do cruzamento.

Um outro tipo de cruzamento considerado útil é o chamado cruzamento uniforme, onde é tomada uma decisão a cada posição do *bit*, se é trocado ou não o *bit* naquele ponto. Por exemplo, uma probabilidade de 60% produz resultados

idênticos a uma taxa de 40%. Caso a taxa seja de 100%, as duas *strings* simplesmente irão trocar de lugar, e caso seja 0% não mudarão.

- 2) Probabilidade de ocorrência: em geral, a probabilidade de ocorrência está entre 60 a 80% para o cruzamento de 1 ponto. Para o cruzamento uniforme, a probabilidade começa alta e vai abaixando até o final da execução terminando com um valor que é metade ou 2/3 do valor inicial.

Quanto maior for esta taxa, mais rapidamente novas estruturas serão introduzidas na população. Mas se esta taxa for muito alta, a maior parte da população será substituída podendo ocorrer perda de estruturas de alta aptidão. Com um valor baixo, o algoritmo pode tornar-se muito lento.

#### **2.4.6 Mutação**

A mutação é a mudança aleatória de alguns *bits* na população. É feita *bit a bit* na população inteira. Geralmente, se utiliza uma taxa de mutação pequena por se tratar de um operador genético secundário. A probabilidade de mutação é freqüentemente mantida constante durante a execução inteira do algoritmo genético, a menos que não produza ótimos resultados (quando a variabilidade da função de avaliação se tornar baixa). Pode ser variável durante a execução e caso varie normalmente é aumentada. Por exemplo, a taxa de mutação pode começar em .001 e finalizar em .01 ou quando o número especificado de gerações estiver completo.

Uma baixa taxa de mutação previne que uma dada posição fique estagnada em um valor, além de possibilitar que se chegue em qualquer ponto do espaço de busca. Com uma taxa muito alta a busca se torna essencialmente aleatória.

O operador de mutação é necessário para a introdução e manutenção da diversidade genética da população, alterando arbitrariamente um ou mais componentes de uma estrutura escolhida, fornecendo assim, meios para introdução de novos elementos na população. Desta forma, a mutação assegura que a probabilidade de se chegar a qualquer ponto do espaço de busca nunca será zero, além de contornar o problema de mínimos locais, pois com este mecanismo, altera-se levemente a direção da busca.

### **2.4.7 Tamanho da População**

O tamanho da população afeta o desempenho global e a eficiência dos Algoritmos Genéticos. Com uma população pequena o desempenho pode cair, pois deste modo a população fornece uma pequena cobertura do espaço de busca do problema. Uma grande população geralmente fornece uma cobertura representativa do domínio do problema, além de prevenir convergências prematuras para soluções locais ao invés de globais. No entanto, para se trabalhar com grandes populações, são necessários maiores recursos computacionais, ou que o algoritmo trabalhe por um período de tempo muito maior.

## Capítulo 3

### Descrição da Sequência de Execução do Trabalho

#### 3.1. Base do Trabalho

Para a execução deste trabalho foi utilizado o Pacote Computacional para o Ensino da Lógica Difusa [2] com o Módulo de Treinamento Genético desenvolvido para o ajuste automático das funções de pertinência [1].

O Pacote Computacional para o Ensino da Lógica Difusa [2] é um software cujo objetivo é estacionar um veículo em uma garagem, partindo de qualquer posição inicial. A Figura 3.1 ilustra a tela principal do programa com os limites das variáveis (x, y e ângulo do carro). A trajetória do veículo é definida pelo usuário através de um conjunto de regras de controle difuso e um conjunto de funções de pertinência desenvolvidos por ele. A Figura 3.2 ilustra um exemplo de um conjunto de funções de pertinência desenvolvido e a Figura 3.3 mostra uma tela com parte das regras difusas editadas. Após a definição das regras e das funções de pertinência, o usuário poderá fazer simulações para verificar a qualidade do controle [9]. Caso o usuário queira minimizar a trajetória do veículo, ele terá que modificar as regras e/ou as funções de pertinência.

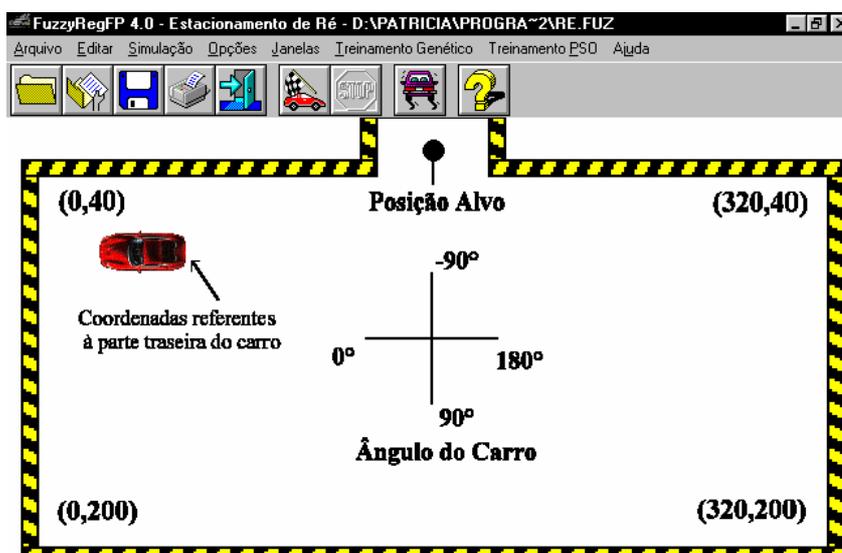


Figura 3.1 – Tela principal do programa

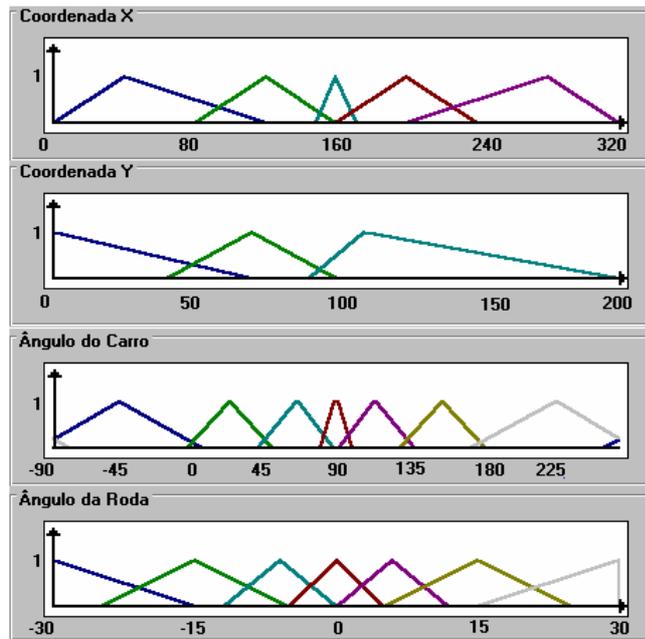


Figura 3.2 – Conjunto de funções de pertinência

**Editar Regras**

Ângulo do Carro  
Corrente :  [←] [→]

Direção  
 Frente  Trás

	ME	PE	CE	PD	MD
TOPO	T_D	T_D	P_D	P_E	T_E
ACIMA	T_D	T_D	T_D	P_E	T_E
CENTRO	T_D	T_D	T_D	P_E	T_E
ABAIXO	T_D	T_D	T_D	P_E	T_E
FUNDO	T_D	T_D	T_D	P_E	T_E

Ângulo da Roda  
Corrente :  **Inverter**  
T\_E  
P\_E  
ZERO  
P\_D  
T\_D

Modificado :

Linguagem Natural  
SE X = ME  
Y = TOPO  
Ângulo = N  
Direção = Frente  
ENTÃO Ângulo da Roda = T\_D .

[Ok] [Ajuda]  
[Cancelar] [Restaurar]

Figura 3.3 – Parte das regras difusas editadas pelo usuário

O Módulo de Treinamento Genético foi feito com o objetivo de ajustar automaticamente as funções de pertinência, utilizando Algoritmos Genéticos, de um controle previamente criado pelo usuário. Os Algoritmos Genéticos são utilizados para achar os melhores parâmetros para as funções de pertinência. Esses parâmetros são os coeficientes de ajuste ( $k_i$  e  $w_i$ ) que fazem com que a função de pertinência se mova para a direita/esquerda ( $k_i$ ) ou se encolha/expandia ( $w_i$ ). A concatenação destes parâmetros constitui os indivíduos da população (cromossomos) que são

avaliados de acordo com o número de iterações necessárias para o veículo estacionar de acordo com o ajuste feito por cada indivíduo.

Para exemplificar como é feito o ajuste de uma função de pertinência, tem-se que a mesma é descrita pelos parâmetros: *IE* (inferior esquerdo), *SE* (superior esquerdo), *SD* (superior direito) e *ID* (inferior direito) [10]. As equações para o ajuste das funções de pertinência são:

$$IE' = (IE + k_i) - w_i$$

$$SE' = (SE + k_i)$$

$$SD' = (SD + k_i)$$

$$ID' = (ID + k_i) + w_i$$

Supondo que os valores da função de pertinência fossem *IE*=30, *SE*=80, *SD*=110 e *ID*=160 e que os coeficientes de ajuste fossem  $k_i = -8$  e  $w_i = 3$ . A função de pertinência seria ajustada conforme a Figura 3.4, substituindo-se os valores nas equações acima descritas.

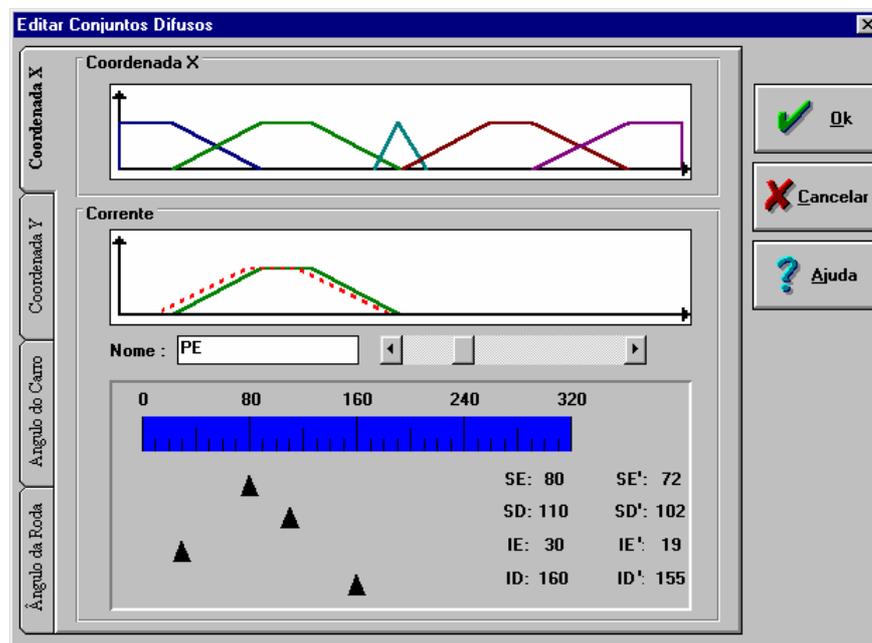


Figura 3.4 – Exemplo de ajuste de uma função de pertinência

Para melhorar ainda mais o Pacote, foi inserido no Módulo de Treinamento Genético o menu Treinamento-Regras, que possui a opção de treinar somente as Regras ou treinar as Regras juntamente com as funções de pertinência. Essas opções serão explicadas em detalhes adiante.

## 3.2. Regras

Para o Treinamento das Regras pôde-se aproveitar o algoritmo desenvolvido por [1], pois como se trata de um algoritmo robusto, era necessário um mínimo de ajustes e o resto do programa não precisaria ser mudado. A série de operações seguidas na implementação de um modelo de Algoritmos Genéticos não muda e essa seqüência pode ser aproveitada num novo programa.

Os componentes genéticos Função de Avaliação, Seleção, Cruzamento e Mutação não mudariam pelo fato de poder-se aproveitar a sua essência. O que mudaria primordialmente seria o cromossomo e depois a maneira de inicializar a população (que não poderia ser aleatória neste caso). Cada um dos componentes genéticos será discutido adiante.

### 3.2.1. Componentes dos Algoritmos Genéticos

Serão apresentados agora os mecanismos utilizados para a geração de regras ótimas no contexto do controlador difuso utilizado através dos Algoritmos Genéticos.

Para isto foi determinado que cada cromossomo seria formado como um vetor linear constituído pelos valores do conjunto de regras do controlador difuso implementado pelo pacote computacional.

Cada regra significa uma ordem dada ao carro em determinada posição. Por exemplo, para  $x=ME$ ,  $y=TOPO$ ,  $\text{ângulo}=L$  e  $\text{direção}=frente$ , foi selecionado o valor  $P\_D$  como  $\text{ângulo da roda}$ , o que corresponde a regra:

SE  $x$  é *ME* E  $y$  é *TOPO* E  $\text{ângulo do carro}$  é *L* E  $\text{sentido do movimento}$  é *PARA FRENTE*  
ENTÃO  $\text{ângulo modificado}$  é  $P\_D$ .

Internamente o valor de cada regra ( $P\_D, T\_D, ZERO, P\_E, T\_E$ ) é representado por um número inteiro. A regra Inverter é representada pelo número  $-1$  e quando não for definida nenhuma regra para determinada posição o valor armazenado é  $-2$ .

Para a formação dos cromossomos das regras são substituídos alguns valores das regras originais por outros segundo uma lógica que será explicada mais adiante.

### 3.2.2. Representação das Variáveis

Um cromossomo é representado por um vetor composto de genes, onde cada cromossomo representa uma solução provável do problema.

Para representar um cromossomo existem diversas maneiras. As mais conhecidas são: a representação binária e a representação por inteiros. Neste trabalho utilizou-se a representação por inteiros uma vez que os genes de cada cromossomo são compostos pelos valores de cada regra que são valores inteiros. Cada retângulo da matriz da Figura 3.5 representa um gene.

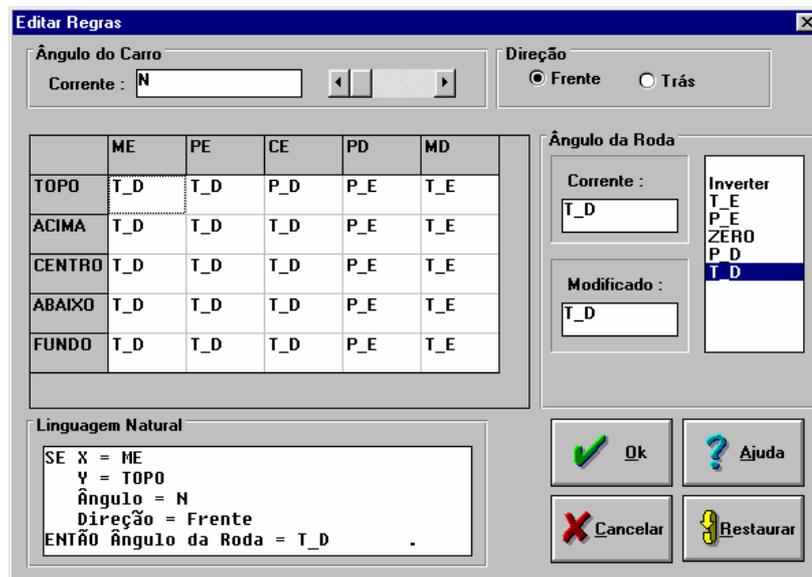


Figura 3.5 – Janela de edição de regras.

Com relação ao tamanho do cromossomo, ou seja, quantos genes cada cromossomo irá ter, isso dependerá do número de funções de pertinência definidas pelo usuário. Para um controle difuso com um grupo de 18 funções de pertinência sendo 6 para x, 5 para y e 7 para o ângulo do carro por exemplo, teremos cromossomos com 420 genes. Isso porque para cada regra tem-se a opção da direção do carro: para frente ou para trás. O cromossomo então é representado por um vetor de 420 posições.

### 3.2.3. Inicialização da População

O processo de geração da população inicial para o problema em questão é muito simples. Utilizou-se para este fim um procedimento condicional.

O primeiro cromossomo gerado possui todos os genes iguais aos valores da regra original e os cromossomos restantes são inicializados alterando-se alguns valores da regra original levando-se em conta as curvas criadas pelo usuário para o ângulo da roda. Por exemplo: Supondo que o usuário definiu para os ângulos da roda 7 curvas. Então se tem a possibilidade de resposta para a tabela de regras 9 valores: sendo os 7 primeiros (0,1,2,3,4,5,6) representando as curvas do ângulo da roda e os restantes as respostas Inverter direção e Sem Regra.

Um novo cromossomo poderia ser obtido trocando-se todas as respostas iguais a 2 por 3, o que significa que uma regra que mandaria o carro girar a roda pouco a esquerda (P\_E), mandaria agora girar totalmente a esquerda (T\_E). Esta mudança não altera por completo a lógica pensada pelo usuário e sim faz um leve ajuste para tentar-se obter melhores resultados. A Figura 3.6 ilustra esta mudança.

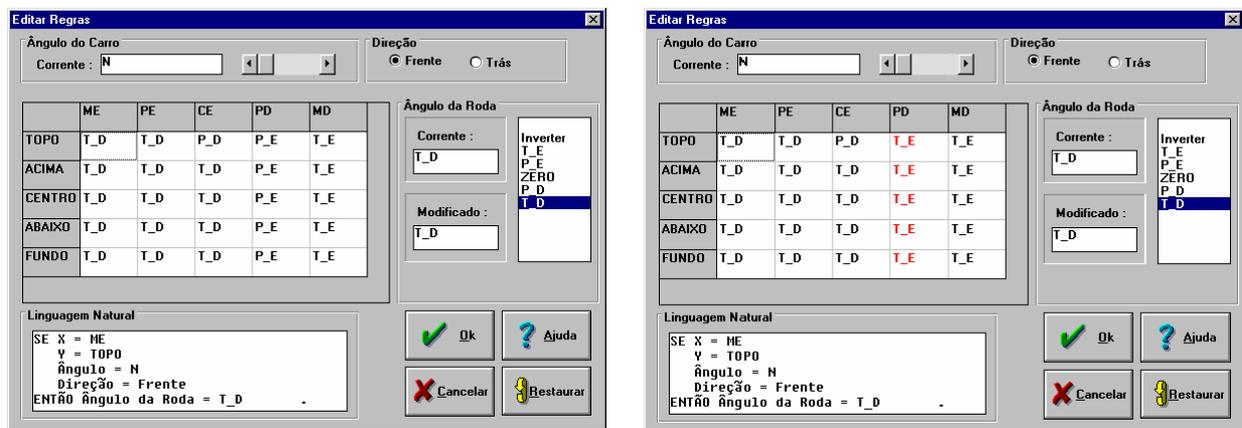


Figura 3.6 – Criação de um novo cromossomo a partir do cromossomo da regra original

Quanto maior for a população especificada, maiores serão as possibilidades de encontrar um ótimo resultado. Foram pré-determinadas algumas alterações de respostas e os demais cromossomos serão formados trocando-se alguns valores por outros aleatórios em 10% dos genes. Estes valores aleatórios são gerados dentro da faixa estabelecida (neste caso, 9 valores).

### 3.2.4. Função de Avaliação

Esta função avalia o nível de adaptação de cada cromossomo gerado pelos algoritmos. Para este trabalho, o objetivo é minimizar a trajetória do veículo até estacionar. Então, a aptidão de cada cromossomo será inversamente proporcional ao número de iterações, ou seja, quanto maior o número de iterações utilizado para o veículo estacionar na garagem, menos apto este indivíduo será e vice-versa.

A função de avaliação é dada por:

$$FA = \frac{1}{1+I}$$

onde  $I$  é o total de iterações até estacionar com base no ajuste feito por cada cromossomo nas regras e nas funções de pertinência.

### 3.2.5. Cruzamento

O processo de cruzamento ou recombinação envolve um corte aleatório que será efetuado nos cromossomos pais que então serão trocados os genes gerando dois descendentes. Considere por exemplo, dois cromossomos pais. Um ponto de cruzamento é escolhido aleatoriamente. As informações anteriores a este ponto em um dos pais são ligadas às informações posteriores à este ponto no outro pai, ilustrado pela Figura 3.7.

Pelo fato dos valores fuzzificados dos ângulos da roda mudarem só um por cromossomo, na hora do cruzamento os cromossomos serem do mesmo tamanho e do ponto de cruzamento ser realizado em pontos semelhantes dos dois cromossomos, a regra filha final não estará muito diferente da regra lógica pensada pelo usuário. Será levemente diferente, ou seja, a regra não será danificada.

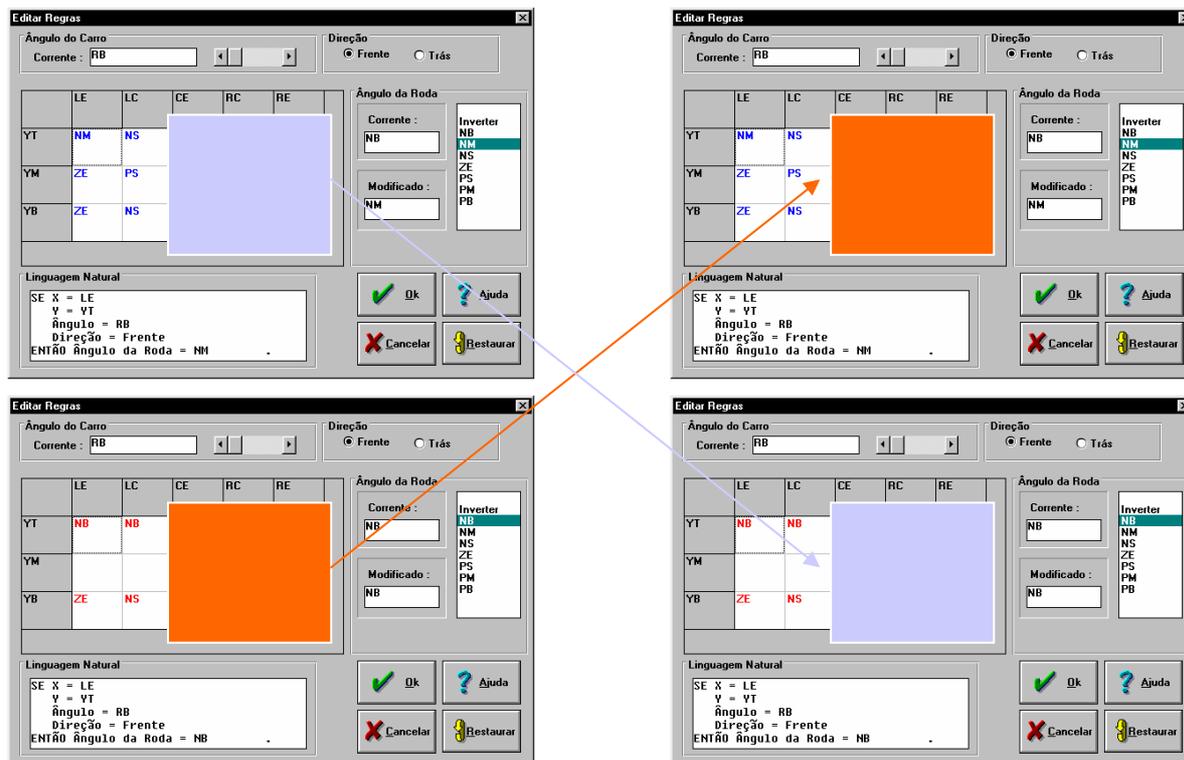


Figura 3.7 – Cruzamento efetuado nas regras

### 3.2.6. Mutação

O processo de mutação consiste na realização de alterações nos valores de um ou mais genes do cromossomo de acordo com a taxa de mutação definida pelo usuário. Para uma taxa igual a 2 por exemplo, ocorrerão alterações em 2% dos genes do cromossomo. A Figura 3.8 ilustra uma mutação ocorrida em um cromossomo.

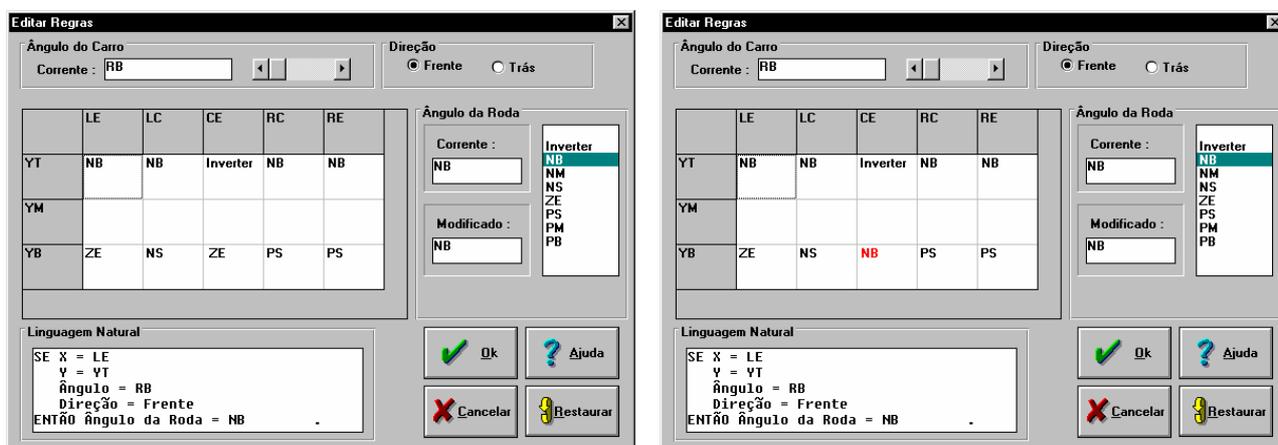


Figura 3.8 – Exemplo de uma mutação ocorrida no cromossomo das regras

### 3.2.7. Seleção através da Roleta da Sorte

O critério de renovação e seleção implementado neste trabalho pelos algoritmos foi o da reprodução. A reprodução é um processo no qual são copiados cromossomos para a próxima geração de acordo com os valores da função de avaliação. Cromossomos com alto valor de aptidão contribuirão com um ou mais descendentes exatamente iguais para a próxima geração.

Considere o exemplo da Tabela 3.1 que mostra uma população  $N = 4$  com o valor da função de avaliação de cada cromossomo igual a  $f_1 = 169$ ,  $f_2 = 576$ ,  $f_3 = 64$  e  $f_4 = 361$ . Como o somatório de todas as funções de avaliação é dado por  $\sum f_i = 1170$ , a aptidão parcial de cada cromossomo é igual a  $169/1170 = 0,14$  (14%),  $576/1170 = 0,49$  (49%),  $64/1170 = 0,06$  (6%) e  $361/1170 = 0,31$  (31%) respectivamente. A partir da aptidão parcial é calculado o número de descendentes esperados de cada cromossomo na próxima geração. No exemplo, para cada cromossomo espera-se  $0,14 \times 4 = 0,56$ ,  $0,49 \times 4 = 1,96$ ,  $0,06 \times 4 = 0,24$  e  $0,31 \times 4 = 1,24$  respectivamente.

O número de cromossomos efetivamente reproduzidos na próxima geração é dado pela parte inteira do número de descendentes esperados de cada cromossomo. Então para o exemplo tem-se uma reprodução de  $x_2$  e uma reprodução de  $x_4$ . A seleção de mais dois cromossomos para a reprodução para completar a população de 4 cromossomos, é feita através da seleção proporcional ao ajustamento.

Tabela 3.1 – Critério de seleção.

População	$f_i(x)$ Avaliação	Aptidão Parcial (%)	Descendentes Esperados	Cromossomos Reproduzidos
$x_1$	169	14	0,56	0
$x_2$	576	49	1,96	1
$x_3$	64	6	0,24	0
$x_4$	361	31	1,24	1
Soma	1170	100	4	2

Este processo foi implementado através da técnica da roleta onde cromossomos que apresentam maior adaptação possuem maiores probabilidades de serem selecionados.

A roleta para este exemplo é representada pela Figura 3.2.

Na prática a roleta é representada por um vetor  $v$  de  $M$  elementos (ordenados de  $\{1, \dots, M\}$  e seja um índice aleatório  $r$ ,  $r = 1, \dots, M$ . Então,  $v(r)$  corresponde a que cromossomo  $i$  foi selecionado. Por exemplo: Com  $M = 10$ , para o exemplo anterior tem-se,  $v = \{1, 2, 2, 2, 2, 2, 3, 4, 4\}$ . Se  $r = 5$ , então o cromossomo selecionado é o de número 2.

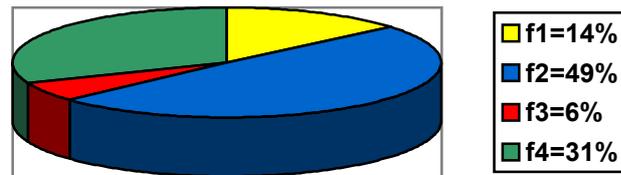


Figura 3.9 – Representação da roleta.

Finalmente, a roleta é girada um determinado número de vezes, dependendo do número de cromossomos que faltam para completar a população, e são escolhidos, como cromossomos que participarão da próxima geração, aqueles sorteados na roleta.

### 3.2.8. Tamanho da População

Para efeito de comparação do desempenho do programa de otimização das regras com o programa original de otimização de funções de pertinência, o tamanho da população foi o mesmo utilizado em [1].

### 3.2.9. Critério de Parada

O critério de parada utilizado foi o da definição do número máximo de gerações a serem produzidas. Quando o número de geração é completado pelos algoritmos genéticos, o processo de geração de novas populações é terminado e a melhor solução é aquela dentre os indivíduos que mais se adaptam à função de avaliação.

### **3.3. O Cromossomo**

No primeiro momento do trabalho, o esforço foi direcionado para ver como seria o cromossomo. Foram utilizados para este fim os artigos [11], [12] e [13] juntamente com os livros [14] e [15]. Com os artigos [11] e [12] foi visto que o cromossomo seria o próprio conjunto de regras e o artigo [13], referenciado pelo livro [14] mostrou a idéia dos operadores de implicação na operação com as matrizes o que levou ao estudo mais profundo destes operadores no livro [15] e nos artigos [16] e [17]. Após estudos de como seria a melhor maneira de substituir o cromossomo de [1] pelo cromossomo de conjunto de regras, concluiu-se que o mesmo seria um cromossomo cujos valores transladariam de um  $n$  para um  $n+1$  ou  $n-1$ . A matriz de regras tem como resposta os conjuntos difusos do ângulo da roda, representados por números que variam de acordo com o número de funções de pertinência usadas para o ângulo da roda (no exemplo utilizado são em número de 7 curvas), portanto variam de um  $\phi_1$  até um  $\phi_7$ . Então os  $\phi$ 's mudariam para um  $\phi_{n-1}$  ou  $\phi_{n+1}$ . Foi feito um teste com esta variação que utilizava a mudança de todos os  $\phi$ 's para  $\phi_{n-1}$ , mas isto alterava muito a regra original e não demonstrava bons resultados. Com isso pensou-se num procedimento condicional que não alterasse em muito a regra, mas sim alterasse a regra pouco a pouco até obter-se a melhor delas. Houve melhora sempre.

### **3.4. Regras juntamente com as Funções de Pertinência (FP's)**

Como em [1] a otimização de FP's sempre melhora a trajetória do veículo e com a inserção da otimização das Regras também ocorrem melhorias, pensou-se numa junção de execução do programa que executasse as duas melhorias de uma só vez, mostrando como resposta uma trajetória do veículo otimizada das duas maneiras de uma só vez, com as melhores curvas possíveis e as melhores regras possíveis, tudo junto. Então, a Janela de Treinamento de Regras foi modificada incorporando-se a ela a opção de Treinamento das FP's (ou seja, ao rodar o treinamento das Regras, o programa das FP's rodaria dentro do programa das Regras, o programa das Regras chamaria a execução do programa das FP's). Para que não houvesse conflito foram criadas novas variáveis (por exemplo, MelhorFPGeração, MelhorFP) e dois cromossomos diferentes (um para as regras e outro para as FP's). Cada um dos dois executa o

Cruzamento, Mutação e Seleção da mesma maneira, seguindo o mesmo modelo matemático; contudo são executados distintamente, usando o cromossomo de regras para o treinamento das regras e o cromossomo das FP's para o treinamento das FP's. Com isso, o programa se tornou muito lento, pois a cada nova regra encontrada é executada para esta regra todas as possibilidades de melhores FP's, para cada posição escolhida do carrinho. Por exemplo, supondo que sejam utilizadas 7 posições de otimização do carrinho. E que o número de população seja 4 e o de geração seja 3. Então, para a regra0 serão geradas 4 FP's. Para achar a FP0 melhor, esta será executada 7 vezes, para achar a FP1 melhor, será executada 7 vezes, e assim até que se chegue a FP4, também executada 7 vezes. E assim segue: para a regra1 ocorre o mesmo cálculo até que se chegue a regra4. E tudo isto para a geração0! O processo se repete para a geração1 e depois para a geração2. O resultado serão as 3 melhores regras encontradas com as melhores FP's para cada uma destas regras. As variáveis MelhorFPGeração e MelhorRegra são muito necessárias para que não se perca a FP respectiva para aquela regra, e deverão ser comparadas com toda a população para ver qual valor possui o menor número de iterações para que a trajetória do veículo seja a menor possível.

### 3.5. Testes

Para efeito de comparação com os resultados obtidos de [1], foram iniciados os testes com os controles GEN1 e GEN2 utilizando o Treinamento com as Regras e o Treinamento com as Regras + FP's.

Dos testes feitos com poucas gerações foi observado que devido às FP's possuírem valores aleatórios de mudança, de repente ocorre um bom valor de número de iterações, enquanto que com os valores de número de iterações obtido das regras são obtidos valores mais padronizados, com números parecidos desde o início. Conclui-se com isto que pelo fato das regras serem lógicas, as mudanças feitas na matriz de regras não alteram muito a regra, enquanto que com as FP's pelo fato de poderem ser aleatórias por tratar-se de variações numéricas nas curvas, de repente uma curva é muito boa em comparação com a curva pensada inicialmente, o que dá um valor de melhora do número de iterações.

Os resultados dos testes serão mostrados posteriormente no Capítulo 5.

### **3.6. Roleta Forçada**

Com o objetivo de se obter melhores resultados, foi feito um teste intitulado Método da Roleta Forçada. Este método utiliza a técnica da estratégia elitista da Roleta da Sorte, ou seja, um indivíduo com alta nota de aptidão é copiado para a próxima geração.

Como a regra original é uma regra pensada, lógica, pensou-se que essa seria a regra elite. Antes do emparelhamento dos cromossomos pais para o cruzamento, é feita a seleção dos descendentes esperados para a nova população. Parte dos indivíduos é selecionada pela nota de aptidão e a outra parte é sorteada. O que muda no Método da Roleta Forçada é que não ocorre o sorteio dos cromossomos pela roleta e sim a verificação do número de sorteios a serem feitos e a colocação do cromossomo com a regra elite na piscina de cruzamentos. Então, como na seleção dos indivíduos já vão ser escolhidos os melhores e na roleta é aleatória a escolha (ou seja, pode ser escolhido um indivíduo ruim), força-se a regra elite na roleta para sempre ser ela a escolhida. A roleta não será mais uma roleta aleatória, será uma roleta forçada, pois terá um valor fixo, que é a regra elite.

Com este teste foi constatado que devido a pouca alteração feita na regra e ao ponto de cruzamento ser igual, os resultados se mantiveram.

## Capítulo 4

### Descrição do Módulo Treinamento Genético

#### 4.1. Apresentação das Funcionalidades

Serão apresentadas as janelas e as operações necessárias para a utilização do Módulo Treinamento Genético. Este módulo tem a função de ajustar as regras juntamente com as funções de pertinência, utilizando algoritmos genéticos, de um controle previamente criado pelo usuário. Para isso, foram inseridas as opções: Treinamento-Regras e Melhores Resultados-Regras ao Módulo original, o qual realizava na versão original somente o ajuste das funções de pertinência. Ambas opções serão explicadas adiante. A Figura 4.1 ilustra as cinco opções do Menu Treinamento Genético: Posições Iniciais para o Treinamento, Treinamento-Funções de Pertinência, Treinamento - Regras, Melhores Resultados - FP e Melhores Resultados - Regras.

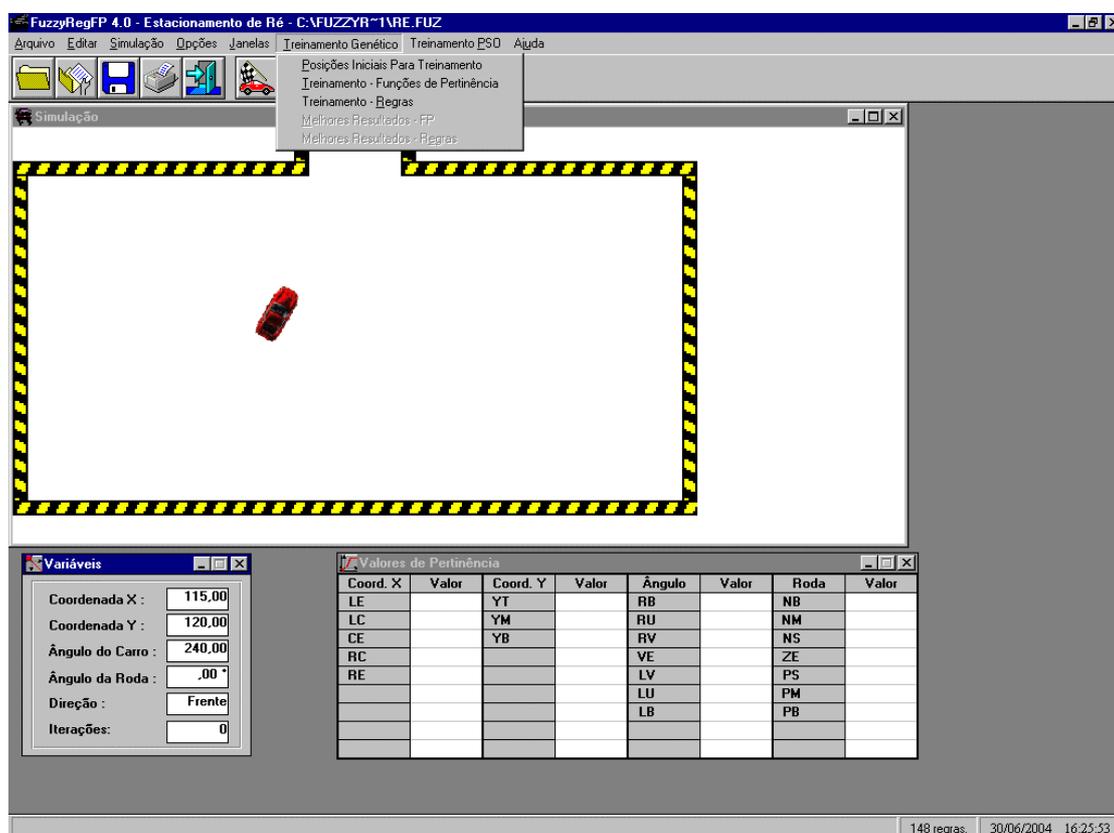


Figura 4.1 – Menu Treinamento Genético

#### 4.1.1. A Opção Definir Posições Iniciais

Para o treinamento genético pode-se definir as posições iniciais que o veículo irá partir para avaliar cada cromossomo que representa o conjunto de valores para os parâmetros das regras e/ou das funções de pertinência, buscando assim uma otimização do controle não somente sobre uma única trajetória, mas sim de todas as posições iniciais possíveis de se partir o veículo para que ocorra o estacionamento.

A Figura 4.2 mostra a janela de edição das posições iniciais. Através de suas opções pode-se definir uma nova posição, editar uma posição existente, excluir e habilitar ou desabilitar uma posição para que esta seja ou não utilizada no treinamento genético. Isso é feito através da opção Utilizar Posição.

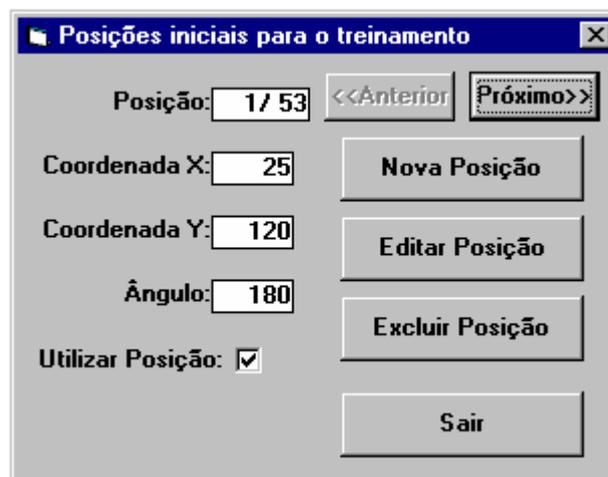


Figura 4.2 – Posições iniciais para treinamento genético

#### 4.1.2. A Opção Treinamento - Regras

Através desta opção o usuário faz o ajuste dos parâmetros para o treinamento genético das regras.

Através da janela mostrada na Figura 4.3, o usuário pode além de definir os parâmetros (população, gerações, taxa de cruzamento e mutação), escolher se serão melhoradas somente as regras ou as regras juntamente com as funções de pertinência. Após iniciar o treinamento, pode-

se acompanhar o treinamento genético a cada população. Para cada cromossomo que constitui o conjunto de parâmetros de ajuste das regras tem-se o número de iterações total geradas para estacionar o veículo partindo de todas as posições iniciais estabelecidas. Outra informação exibida por esta janela é a melhor geração, ou seja, em qual geração temos a melhor minimização de espaço percorrido pelo veículo até estacionar partindo de todas as posições iniciais estabelecidas.

The image shows a software dialog box titled "Parâmetros dos AG's - Regras". It features several input fields and control elements:

- População:** Input field with the value "4".
- Gerações:** Input field with the value "5".
- Crossover:** Input field with the value "90" followed by a percentage sign "%".
- Mutação:** Input field with the value "2" followed by a percentage sign "%".
- Buttons:** A green "Iniciar" button with a checkmark icon and a red "Sair" button with an "X" icon.
- Checkboxes:** Two checkboxes, both currently unchecked: "Treinar Regras + Funções de Pertinência" and "Treinar apenas para a posição atual".
- Output Fields:** Five empty input fields labeled "Geração:", "Cromossomo:", "Iterações:", "Melhor Ger.:", and "Iterações:".
- Bottom Button:** A button labeled "Melhores" centered at the bottom.

Figura 4.3 – Parâmetros genéticos

Concluídas todas as gerações, pode-se escolher o melhor resultado encontrado entre todas as gerações através do botão Melhores. Esta operação permite avaliar as soluções propostas pelos algoritmos genéticos. Para isso o usuário deverá escolher uma geração e clicar no botão Ajustar como mostrado na Figura 4.4.

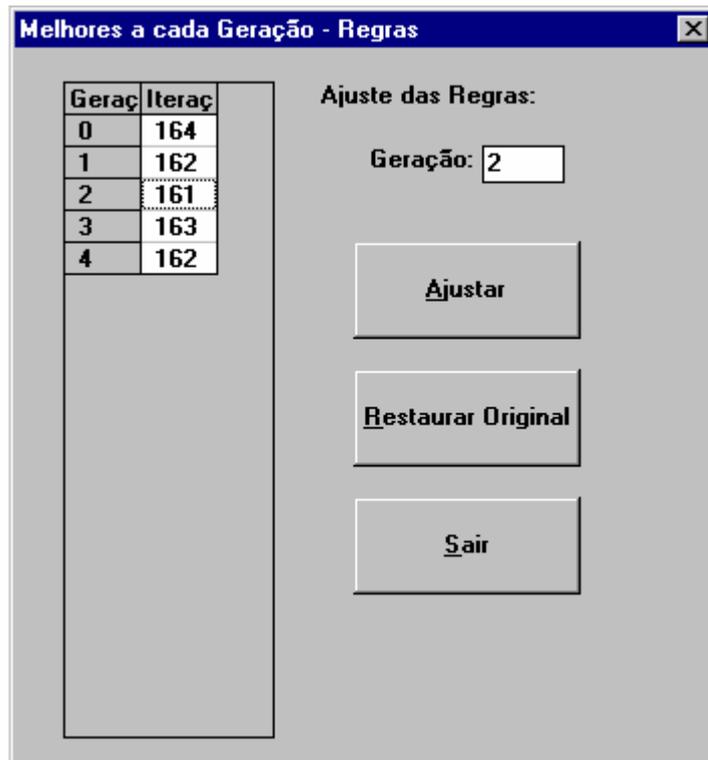


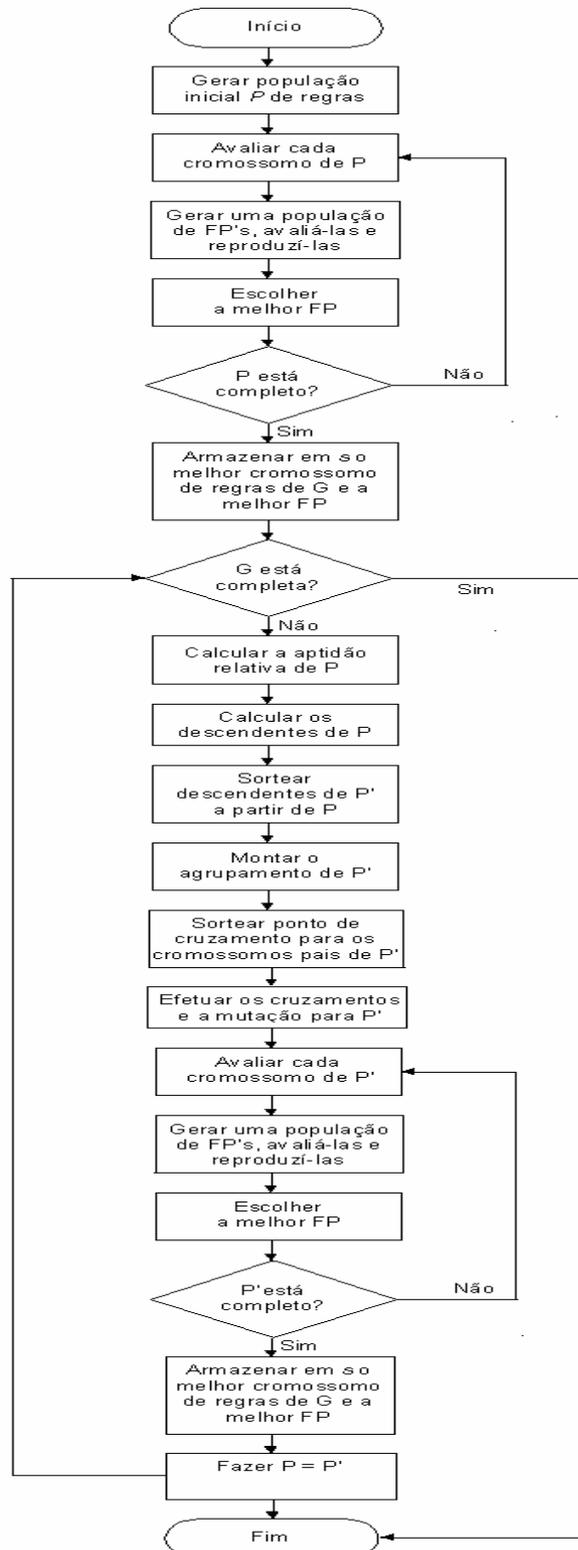
Figura 4.4 – Melhores resultados

Após o ajuste, se foram melhoradas somente as regras, estas são redefinidas segundo os parâmetros do cromossomo correspondente a geração escolhida. O sistema fará o controle com base nestas novas regras. As regras originais podem ser restauradas através da opção Melhores Resultados - Regras no menu Treinamento Genético clicando-se no botão Restaurar Original.

Do mesmo modo, após o ajuste, se foram melhoradas as regras juntamente com as funções de pertinência, ambas são redefinidas segundo os parâmetros dos cromossomos correspondentes a geração escolhida. O sistema fará o controle com base nestas novas regras e funções de pertinência. As regras e as funções de pertinência originais podem ser restauradas através da opção Melhores Resultados - Regras no menu Treinamento Genético clicando-se no botão Restaurar Original.

## 4.2. Apresentação do Algoritmo

Nesta seção o algoritmo será ilustrado através de um fluxograma e de um acompanhamento passo a passo para entender-se de que maneira foi feita a programação da otimização das regras juntamente com as funções de pertinência através dos Algoritmos Genéticos.



Seja  $P$  a população,  $G$  o número de gerações,  $P_c$  a taxa de cruzamento (recombinação) e  $P_m$  a taxa de mutação, o algoritmo abaixo apresentado gera como saída o vetor  $s$  com  $G$  posições. Cada elemento deste vetor é o melhor cromossomo de uma geração.

- Passo 1. Gerar população inicial  $P$  de regras cujos genes representam as saídas fuzzificadas dos ângulos da roda.
- Passo 2. Avaliar cada cromossomo da população  $P$ .
  - Passo 2.1. Gera uma população inicial de funções de pertinência, faz a avaliação e a reprodução destas.
  - Passo 2.2. Escolhe a melhor função de pertinência para o cromossomo de regras que está sendo avaliado.
- Passo 3. Retornar ao Passo 2 até que complete o número de população  $P$ .
- Passo 4. Armazenar no vetor  $s$  o melhor cromossomo de regras encontrado na geração  $G$  e a melhor função de pertinência encontrada para este cromossomo de regras.
- Passo 5. Se completou o número de geração  $G$  vá para o Passo 17.
- Passo 6. Calcular a aptidão relativa da população  $P$ .
- Passo 7. Calcular os descendentes esperados da população  $P$ .
- Passo 8. Sortear descendentes da população  $P'$  a partir da população  $P$ .
  - Passo 8.1 Utilizando o Método da Roleta
  - Passo 8.2 Utilizando o Método da Roleta Forçada
- Passo 9. Montar o agrupamento da população  $P'$
- Passo 10. Sortear ponto de cruzamento para os cromossomos pais da população  $P'$ .
- Passo 11. Efetuar os cruzamentos para a população  $P'$  segundo  $P_c$ .
- Passo 12. Efetuar a mutação em cada cromossomo segundo  $P_m$ .
- Passo 13. Avaliar cada cromossomo da população  $P'$ .
  - Passo 13.1. Gera uma população inicial de funções de pertinência, faz a avaliação e a reprodução destas.
  - Passo 13.2. Escolhe a melhor função de pertinência para o cromossomo de regras que está sendo avaliado.
- Passo 14. Retornar ao Passo 13 até que complete o número de população  $P'$ .
- Passo 15. Armazenar no vetor  $s$  o melhor cromossomo de regras encontrado na geração  $G$  e a melhor função de pertinência encontrada para este cromossomo de regras.

Fazer  $P = P'$ .
- Passo 16. Retornar ao Passo 5.
- Passo 17. Fim

A Figura 4.5 mostra a trajetória feita pelo veículo que parte da coordenada X igual a 40, Y igual a 183 e ângulo do carro igual a  $-46$  gerando 558 iterações até estacionar. O algoritmo será utilizado para minimizar o número de iterações e, conseqüentemente, o espaço percorrido. Para este acompanhamento definiu-se que o algoritmo teria  $G = 5$  gerações e uma população de 4 cromossomos. Taxa de cruzamento  $P_c = 90\%$  e taxa de mutação  $P_m = 2\%$ . A Figura 4.6 mostra as funções de pertinência que serão ajustadas e a Figura 4.7 mostra um retrato das regras que serão modificadas.

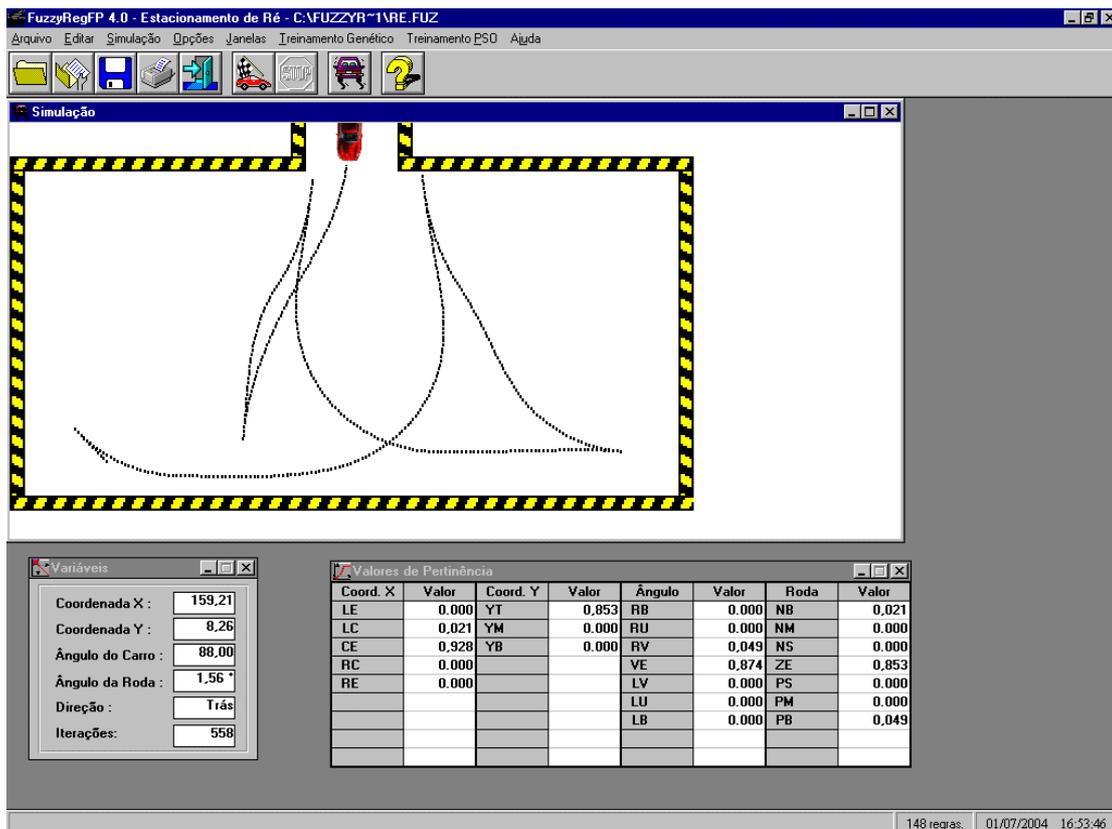


Figura 4.5 – Simulação sem treinamento genético.

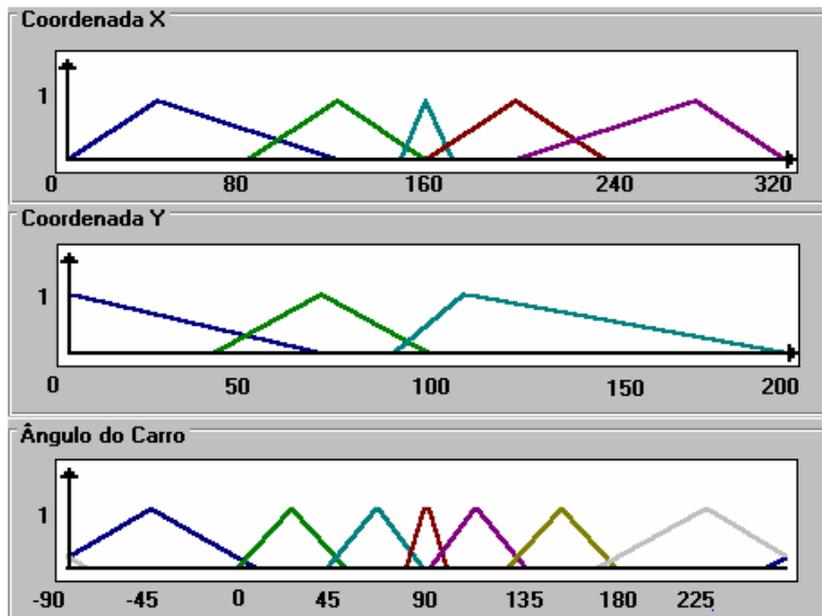


Figura 4.6 – Funções de pertinência sem ajuste.

**Editar Regras**

Ângulo do Carro  
Corrente : VE

Direção  
 Frente  Trás

	LE	LC	CE	RC	RE
YT		NB	ZE	PB	
YM	NB	NB	ZE	PB	PB
YB	NB	NM	ZE	PM	PB

Ângulo da Roda  
Corrente : NB

Modificado : NB

Inverter  
NB  
NM  
NS  
ZE  
PS  
PM  
PB

Linguagem Natural  
SE X = LE  
Y = YM  
Ângulo = VE  
Direção = Trás  
ENTÃO Ângulo da Roda = NB .

Ok Ajuda  
Cancelar Restaurar

Figura 4.7 – Um retrato das regras sem ajuste.

**Passo 1:**

O primeiro passo do algoritmo gera a população inicial  $P$  com genes representando as saídas fuzzificadas dos ângulos da roda, lembrando que o cromossomo inicial é o conjunto original de regras escritas pelo usuário. Para a construção das regras é necessário saber quantas funções de pertinência são utilizadas pelas variáveis de entrada ( $x$ ,  $y$  e ângulo do carro) e levar em conta também a direção do veículo (frente e trás). O controle difuso utilizado neste exemplo possui 15 funções de pertinência, sendo 5 funções de pertinência para  $x$ , 3 funções de pertinência para  $y$  e 7 funções de pertinência para o ângulo do carro. Então cada cromossomo possuirá 210 genes, resultado da multiplicação  $5 \times 3 \times 7 \times 2$ .

**Passo 2:**

No segundo passo do algoritmo a população inicial é avaliada. Esta avaliação consiste em ajustar o conjunto de regras originais de acordo com os valores sugeridos por cada cromossomo.

**Passo 2.1:**

Após ajustar o conjunto de regras com os novos valores, é gerada uma população inicial de funções de pertinência. Estas funções de pertinência são ajustadas e avaliadas junto com este conjunto de regras, ou seja, a melhor função de pertinência é aquela que obteve o menor número de iterações para a geração 0 de funções de pertinência. São escolhidos os cromossomos de FP's para o cruzamento e a mutação criando uma nova geração de funções de pertinência. Este processo é repetido até que sejam executadas todas as gerações de funções de pertinência.

**Passo 2.2:**

É escolhida a melhor FP de todas as gerações de FP's para o cromossomo de regras que está sendo avaliado. Esta FP é guardada num vetor denominado MelhorFP, armazenado na 1ª geração no vetor MelhorFPGeração. Os próximos vetores MelhorFP só serão armazenados se tiverem o número de iterações menor do que o valor do MelhorFPGeração.

**Passo 3:**

Neste passo é verificado se foi completado o número de populações estabelecidas. Caso contrário o algoritmo irá para o Passo 2 até que complete o número de população  $P$ .

#### Passo 4:

O melhor cromossomo de regras juntamente com a melhor FP para este cromossomo são ajustados respectivamente na matriz de regras e nos conjuntos difusos. Após o ajuste o controlador difuso do pacote computacional é executado automaticamente retornando o número de iterações geradas até estacionar. Ao final, o melhor cromossomo de regras e a melhor FP para este cromossomo são armazenados no vetor  $s$ .

#### Passo 5:

Neste passo é verificado se foi completado o número de gerações estabelecidas. Caso afirmativo o algoritmo irá para o Passo 17.

#### Passo 6:

Neste passo calcula-se a aptidão relativa de cada cromossomo da população. Esta aptidão é o valor de  $f_i(x) / \sum f_i(x)$ . Na Tabela 4.1, tem-se a soma das avaliações igual a 1616, resultando em uma aptidão relativa igual a 19,5%, 37%, 6% e 37,5% respectivas a cada cromossomo.

#### Passo 7:

Com os valores da aptidão relativa calcula-se o número de descendentes a serem reproduzidos de cada cromossomo na nova população  $P'$ . Na Tabela 4.1, o número de descendentes de  $x_0$ ,  $x_1$ ,  $x_2$  e  $x_3$  na nova população é igual a 0, 1, 0, 1 respectivamente. Estes valores são a parte inteira da aptidão relativa.

Tabela 4.1 – Primeira geração.

População Inicial	Iterações	Avaliação	Aptidão Parcial (%)	Descendentes Esperados	Seleção	
					Reprodução	Roleta
$x_0$	316	315	19,5	0,78	0	1
$x_1$	167	595	37	1,48	1	0
$x_2$	1000	100	6	0,24	0	1
$x_3$	164	606	37,5	1,5	1	0
<b>Soma</b>	1647	1616	100	4	2	2
<b>Média</b>	411.75	404	25			
<b>Max</b>	1000	595	37,5			

População $P'$	Emparelhamento	Ponto de Cruzamento	Genes selecionados para mutação
$x_0$	1	7	-
$x_2$	0	7	12
$x_1$	3	22	-
$x_2$	2	22	1-16

### Passo 8:

Existem duas opções para o sorteio dos cromossomos. Uma feita através do Método da Roleta (que é o utilizado pelo algoritmo efetivamente) e uma variante feita para efeito de teste para tentar-se obter um melhor resultado chamada de Método da Roleta Forçada.

#### Passo 8.1:

Através da roleta, serão sorteados os cromossomos necessários para se completar a nova população  $P'$ . Na Tabela 4.1 nota-se que dois cromossomos da nova população são sorteados pela roleta. Os cromossomos sorteados são reproduzidos na nova população  $P'$ .

#### Passo 8.2:

Uma outra opção para não alterar muito a lógica feita pelo usuário para as regras seria forçar que os cromossomos necessários para se completar a nova população  $P'$  fossem sempre iguais à regra original. Desta maneira não haveria o sorteio dos cromossomos pela roleta e sim a verificação do número de sorteios a serem feitos e a colocação do cromossomo com a regra original na piscina de cruzamentos.

### Passo 9:

Neste passo é feito o agrupamento para o cruzamento onde os cromossomos pais são emparelhados através de sorteio. Na Tabela 4.1 o emparelhamento ocorre da seguinte forma:  $x_0$  com  $x_1$ ,  $x_2$  com  $x_3$ .

Tabela 4.2 – Segunda geração.

Nova População	Iterações	Avaliação	Aptidão Parcial (%)	Descendentes Esperados	Seleção	
					Reprodução	Roleta
$x_0$	162	613	36	1,44	1	0
$x_1$	508	196	11,5	0,46	0	1
$x_2$	163	610	35,5	1,42	1	1
$x_3$	336	297	17	0,68	0	0
<b>Soma</b>	1169	1716	100	4	2	2
<b>Média</b>	292,25	429				
<b>Max</b>	508	613				

População $P'$	Emparelhameto	Ponto de Cruzamento	Genes selecionados para mutação
$x_0$	3	8	-
$x_2$	2	20	-
$x_1$	1	20	5
$x_2$	0	8	-

**Passo 10:**

Neste passo um ponto de cruzamento é escolhido aleatoriamente para cada par de cromossomos emparelhados como mostrado nas Tabelas 4.1, 4.2, 4.3 e 4.4.

**Passo 11:**

Neste passo é efetuado o cruzamento entre os cromossomos pais segundo a probabilidade de cruzamento estabelecida pelo usuário em  $P_c$ . De acordo com o ponto de cruzamento definido no passo anterior os genes anteriores a este ponto em um dos pais são ligados aos genes posteriores a este ponto no outro pai gerando assim dois novos descendentes.

**Passo 12:**

A mutação consiste em selecionar cada gene e alterá-lo para um valor do intervalo  $-2$  (Inverter Regra) até o número que representa o total de curvas usadas para o ângulo da roda menos 1. Ou seja, no exemplo utilizado o intervalo é de  $[-2,6]$ . A alteração de cada gene acontece segundo a probabilidade de mutação estabelecida em  $P_m$ . Nas Tabelas 4.1, 4.2, 4.3 e 4.4 verifica-se uma baixa ocorrência de mutação em função da baixa probabilidade estabelecida neste exemplo ( $P_m=2\%$ ). Este valor evita que o processo se torne essencialmente aleatório.

**Passo 13:**

Neste passo a nova população  $P'$  é avaliada. A população  $P$  passa a ser igual a  $P'$ . Esta avaliação consiste em ajustar o conjunto de regras originais de acordo com os valores sugeridos por cada cromossomo da população  $P'$ .

**Passo 13.1:**

Após ajustar o conjunto de regras com os novos valores, é gerada uma população inicial de funções de pertinência. Estas funções de pertinência são ajustadas e avaliadas junto com este conjunto de regras, ou seja, a melhor função de pertinência é aquela que obteve o menor número de iterações para a geração 0 de funções de pertinência. São escolhidos os cromossomos de FP's para o cruzamento e a mutação criando uma nova geração de funções de pertinência. Este processo é repetido até que sejam executadas todas as gerações de funções de pertinência.

**Passo 13.2:**

É escolhida a melhor FP de todas as gerações de FP's para o cromossomo de regras que

está sendo avaliado. Esta FP é guardada num vetor denominado MelhorFP, armazenado na 1ª geração no vetor MelhorFPGeração. Os próximos vetores MelhorFP só serão armazenados se tiverem o número de iterações menor do que o valor do MelhorFPGeração.

Tabela 4.3 – Terceira geração.

Nova População	Iterações	Avaliação	Aptidão Parcial (%)	Descendentes Esperados	Seleção	
					Reprodução	Roleta
$x_0$	317	314	17	0,68	0	0
$x_1$	296	337	18	0,72	0	1
$x_2$	164	606	32	1,28	1	1
$x_3$	161	617	33	1,32	1	0
Soma	938	1874	100	4	2	2
Média	234,5	468,5				
Max	317	617				

População P'	Emparelhamento	Ponto de Cruzamento	Genes selecionados para mutação.
$x_1$	3	5	-
$x_2$	2	27	-
$x_3$	1	27	-
$x_2$	0	5	4 - 12

**Passo 14:**

Retornar ao Passo 13 até que complete o número de população P'.

**Passo 15:**

O melhor cromossomo de regras juntamente com a melhor FP para este cromossomo são ajustados respectivamente na matriz de regras e nos conjuntos difusos. Após o ajuste o controlador difuso do pacote computacional é executado automaticamente retornando o número de iterações geradas até estacionar. Ao final, o melhor cromossomo de regras e a melhor FP para este cromossomo são armazenados no vetor  $s$ .

**Passo 16:**

Retornar ao Passo 5.

**Passo 17:**

Neste passo o algoritmo se encerra. Na Tabela 4.4 são mostrados a avaliação da última geração e o número de iterações geradas. Os melhores cromossomos de cada geração são listados para que o usuário escolha a geração com o menor número de iterações.

Tabela 4.4 – Última geração.

Nova População	Iterações	Avaliação	Aptidão Parcial (%)	Descendentes Esperados	Seleção	
					Reprodução	Roleta
$x_0$	163	610	33	1,32	1	1
$x_1$	321	310	16	0,64	0	0
$x_2$	163	610	33	1,32	1	1
$x_3$	296	337	18	0,72	0	0
<b>Soma</b>	943	1867	100	4	2	2
<b>Média</b>	235,75	466,75				
<b>Max</b>	296	610				

População P'	Emparelhameto	Ponto de Cruzamento	Genes selecionados para mutação.	Iterações	Avaliação da última geração
$x_0$	2	5	6	162	613
$x_2$	3	17	-	163	610
$x_0$	0	5	-	163	610
$x_2$	1	17	-	162	613
<b>Soma</b>				650	2446
<b>Média</b>				162,5	611,5
<b>Max</b>				163	613

Após a escolha do melhor cromossomo entre todas as gerações será feito o ajuste das regras juntamente com as funções de pertinência do controlador difuso utilizado neste exemplo. Na Figura 4.8 são mostrados os melhores resultados. Na Figura 4.9 temos a simulação feita após o ajuste. A Figura 4.10 mostra uma das telas do conjunto de regras ajustado e a Figura 4.11 mostra as funções de pertinência após o ajuste feito nas funções e implicação das variáveis  $x$ ,  $y$  e ângulo do carro.

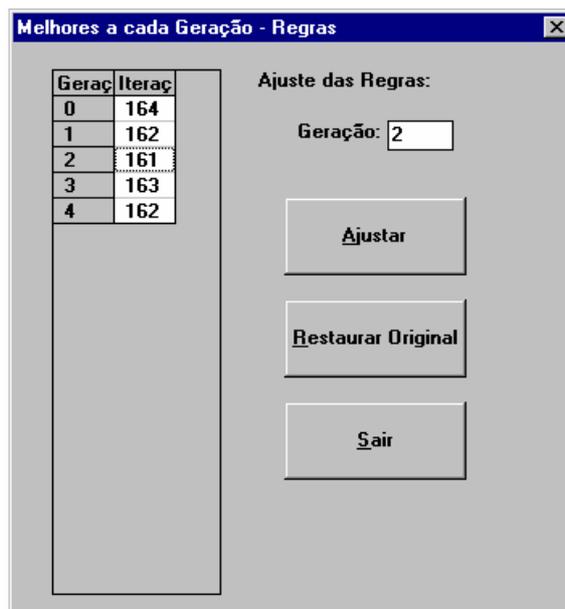


Figura 4.8 – Melhores resultados do treinamento genético.

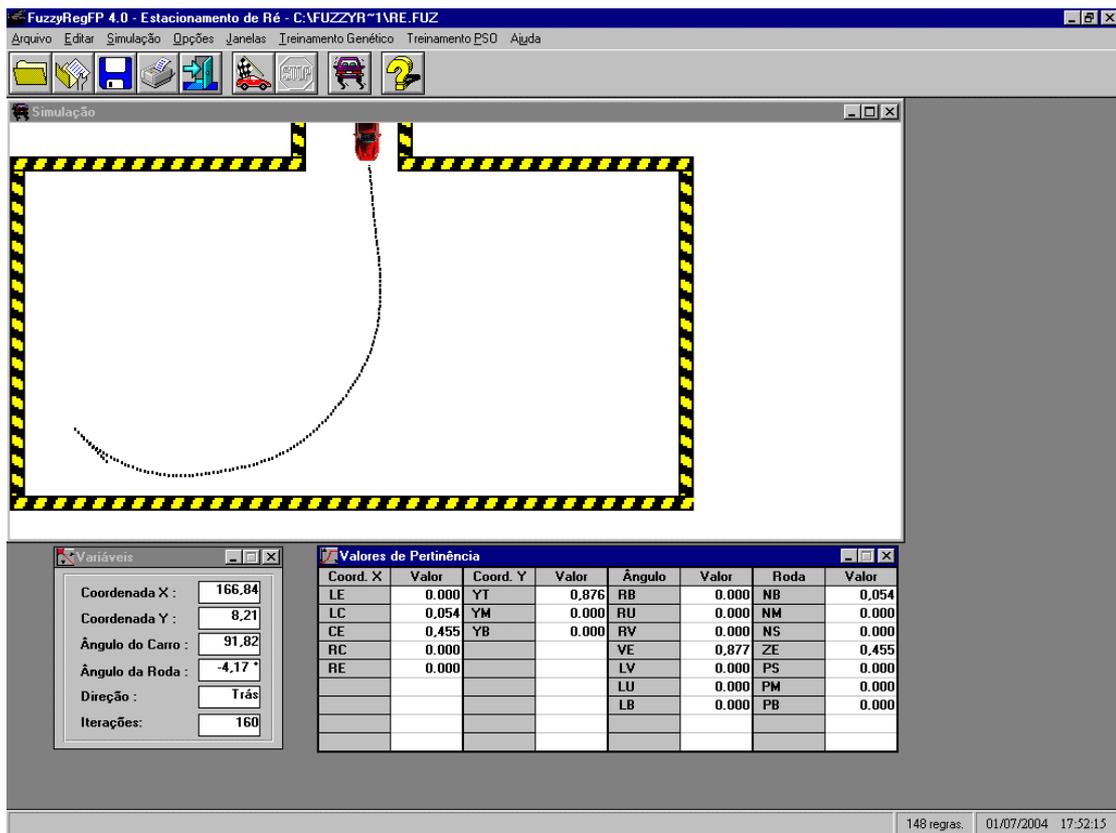


Figura 4.9 – Simulação após treinamento genético.

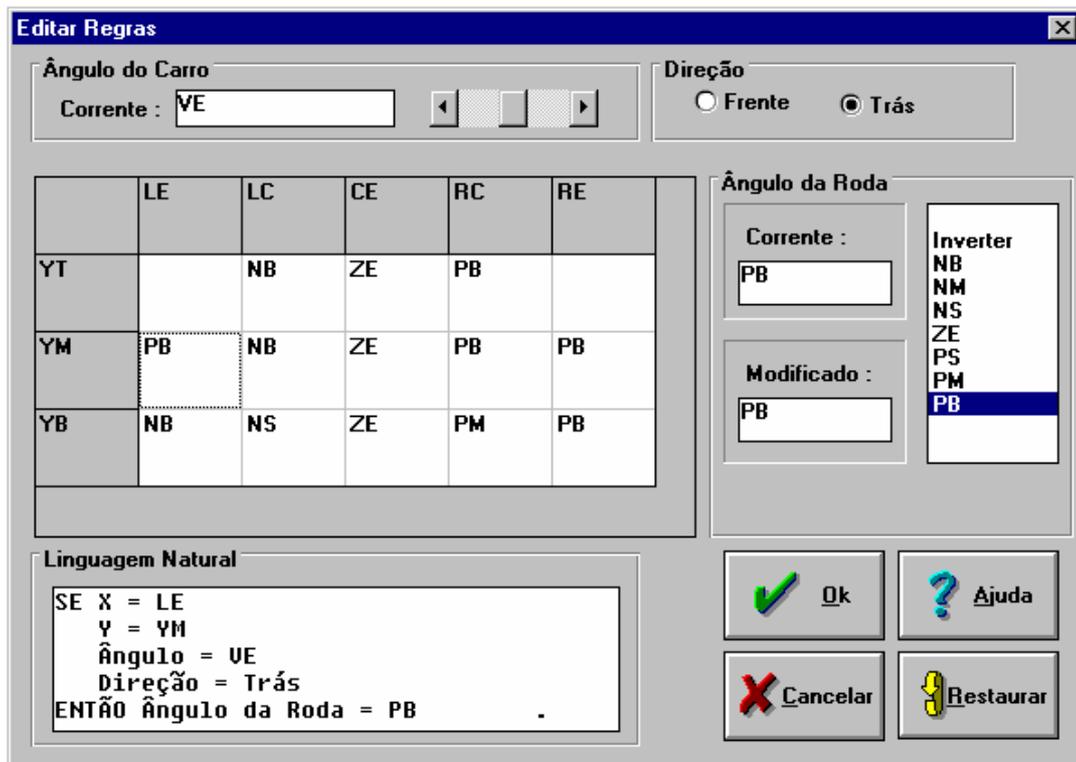


Figura 4.10 – Um retrato das regras após o ajuste

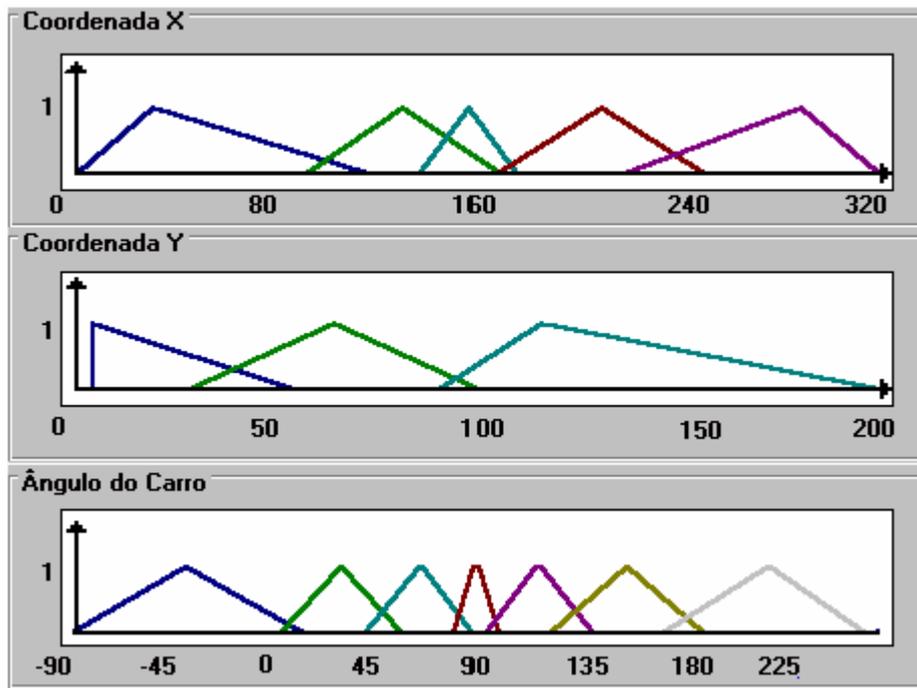


Figura 4.11 – Funções de pertinência após o ajuste.

Nota-se uma sensível redução na trajetória do veículo até estacionar. Com as funções de pertinência originais são necessárias 558 iterações para ocorrer o estacionamento (Figura 4.5) enquanto que após o treinamento genético este número é reduzido para 161 iterações (Figura 4.9). Poder-se-ia obter melhores resultados com uma população maior e com mais gerações no treinamento. No entanto, o intuito neste exemplo é demonstrar o algoritmo passo-a-passo para a compreensão do seu funcionamento. Melhores resultados serão apresentados no próximo capítulo.

# Capítulo 5

## Testes

### 5.1. Introdução

Este capítulo apresenta testes realizados com controles difusos que tiveram suas regras e funções de pertinência ajustadas através dos algoritmos genéticos. Estes testes permitem uma avaliação objetiva dos resultados encontrados.

Foram feitos dois exemplos de ajustes nas regras e funções de pertinência do controle RE gerando dois novos controles: O GEN1 e GEN2. Estes controles possuem 148 regras e 15 funções de pertinência para as variáveis de entrada  $x$ ,  $y$  e ângulo da roda. As funções de pertinência originais são mostradas na Figura 5.1 e as regras originais se encontram no Anexo A. Foram escolhidos os mesmos controles feitos em [1] para efeito de comparação dos resultados.

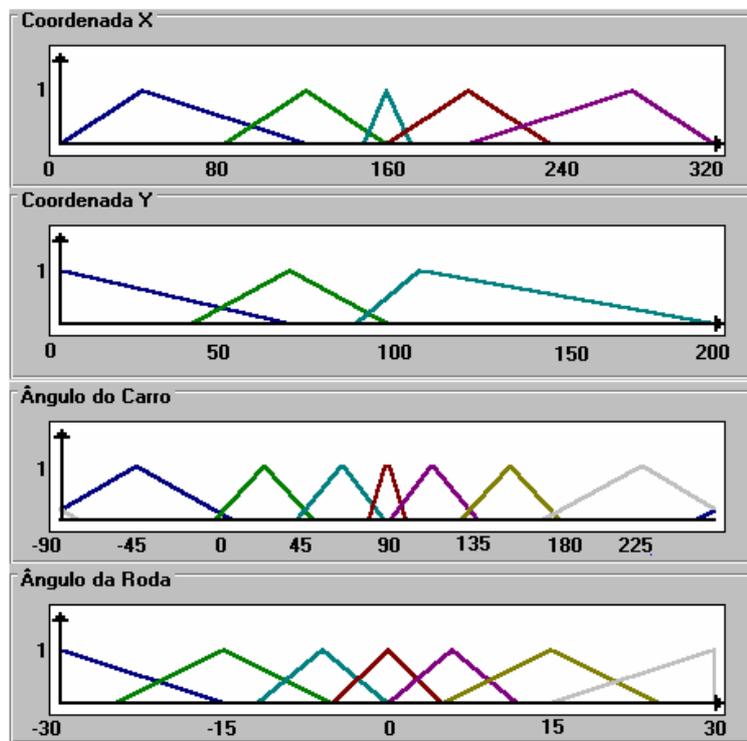


Figura 5.1 – Funções de pertinência do controle RE.

## 5.2. O controle GEN1

O treinamento deste controle foi feito a partir de três posições iniciais conforme mostra a Tabela 5.1. Nesta tabela tem-se também o número de iterações geradas pelo veículo até estacionar utilizando as regras e as funções de pertinência originais.

Tabela 5.1 – Posições iniciais para o controle GEN1.

Posição	X	Y	Ângulo do Carro	Iterações sem treinamento
1	25	120	180	330
2	160	130	-90	888
3	275	160	-40	655

A Figura 5.2 mostra o veículo em cada uma das posições iniciais.

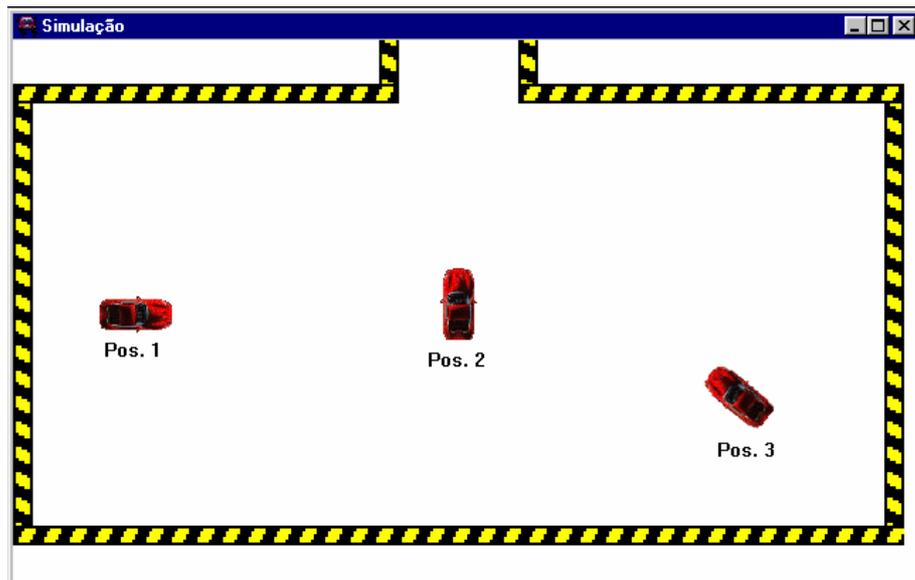


Figura 5.2 – Posições iniciais de treinamento do controle GEN1.

Estas posições foram escolhidas de acordo com pontos onde o veículo não desenvolve uma boa trajetória até estacionar e, conseqüentemente, gerando um número excessivo de iterações. A definição de várias posições iniciais não irá somente minimizar as trajetórias referentes a estes pontos, mas como também para outros pontos, conseguindo assim uma minimização global de espaço percorrido.

A Figura 5.3 mostra as trajetórias referentes a cada posição inicial.

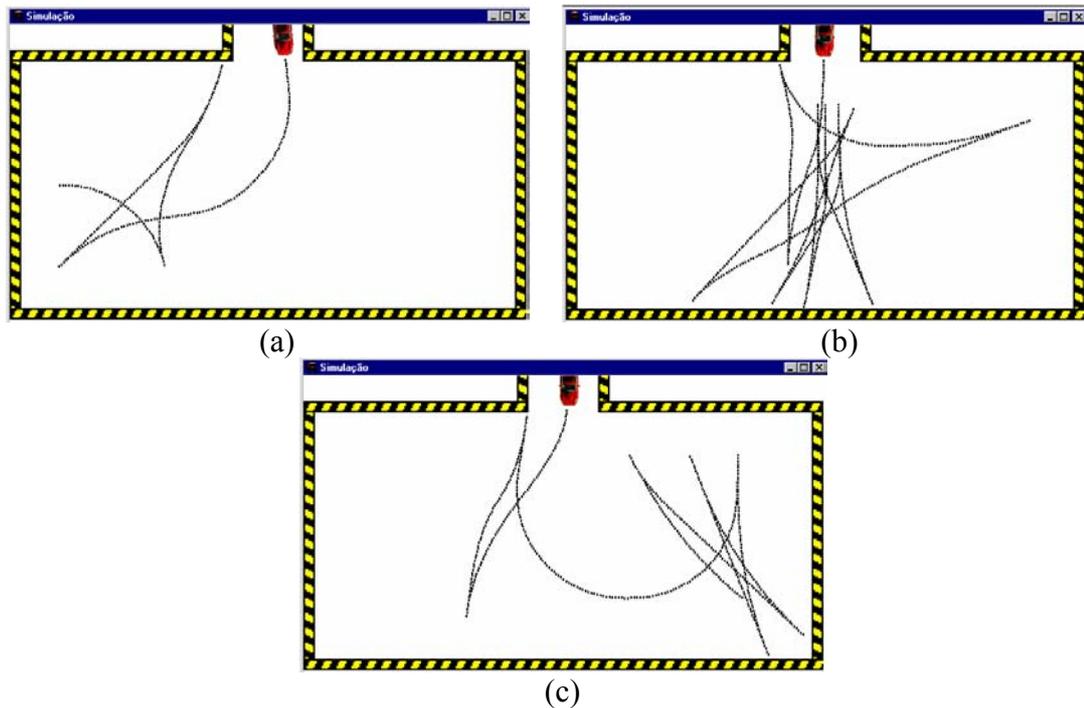


Figura 5.3 – Simulações sem treinamento.  
(a) Posição 1 - (b) Posição 2 – (c) Posição 3

Foram feitos 2 testes diferentes para a verificação da eficiência dos mecanismos utilizados, os quais serão mostrados a seguir.

### 5.2.1. Treinamento de Regras para o controle GEN1

Os parâmetros genéticos definidos para o treinamento são mostrados na Tabela 5.2. Foram escolhidos os mesmos parâmetros de [1] para a comparação dos resultados.

Tabela 5.2 – Parâmetros genéticos para o treinamento de GEN1.

Tamanho da População	14
Número de Gerações	30
Probabilidade de Crossover	90%
Probabilidade de Mutação	1%

O resultado gerado pelos algoritmos genéticos é mostrado na Tabela 5.3. Foi colocado também o resultado obtido em [1] para efeito de comparação dos resultados.

Tabela 5.3 – Iterações após o treinamento genético de GEN1.

Posição	Iterações sem treinamento	Iterações com treinamento de FP's*	Iterações com treinamento de regras
1	330	280	191
2	888	384	344
3	655	277	220
Total	1873	941	755
Média	624,33	331,67	251,66

\*Resultados de [1]

A Figura 5.4 ilustra uma comparação das trajetórias do veículo relativa a cada treinamento feito.

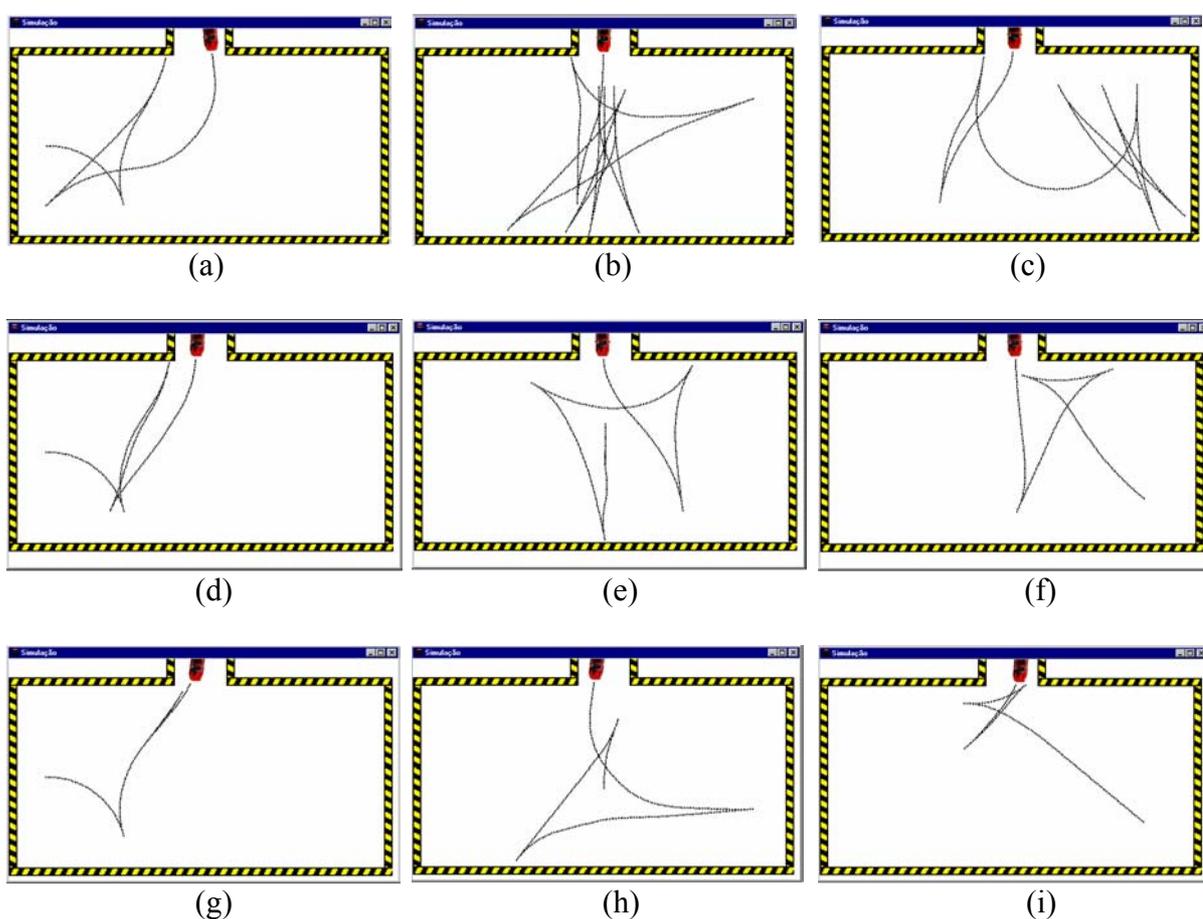


Figura 5.4 – Comparações das trajetórias do treinamento de regras de GEN1.

(a) Posição 1-sem treinamento (b) Posição 2-sem treinamento (c) Posição 3-sem treinamento (d) Posição 1-treinamento de FP's (e) Posição 2-treinamento de FP's (f) Posição 3-treinamento de FP's (g) Posição 1-treinamento de regras (h) Posição 2-treinamento de regras (i) Posição 3-treinamento de regras.

Como mostrado na Tabela 5.3 e ilustrado na Figura 5.4, obteve-se uma redução de 1118 iterações (59,7%) para o veículo estacionar partindo-se das posições

inicias fixadas para o treinamento. No Anexo B encontra-se o conjunto de regras modificado após o ajuste (GEN1 Regras).

Com relação ao executado em [1], o treinamento de regras apresentou melhora quanto ao número de iterações, demonstrando a eficácia do treinamento de regras.

### 5.2.2. Treinamento de Regras + FP's para o controle GEN1

Os parâmetros genéticos definidos para o treinamento são mostrados na Tabela 5.4. Foram escolhidos os mesmos parâmetros de [1] para a comparação dos resultados.

Tabela 5.4 – Parâmetros genéticos para o treinamento de GEN1.

Tamanho da População	14
Número de Gerações	30
Probabilidade de Crossover	90%
Probabilidade de Mutação	1%

O resultado gerado pelos algoritmos genéticos é mostrado na Tabela 5.5. Foi colocado também o resultado obtido em [1] para efeito de comparação dos resultados.

Tabela 5.5 – Iterações após o treinamento genético de GEN1.

Posição	Iterações sem treinamento	Iterações com treinamento de FP's*	Iterações com treinamento de regras + FP's
1	330	280	140
2	888	384	309
3	655	277	218
Total	1873	941	667
Média	624,33	331,67	222,33

\*Resultados de [1]

A Figura 5.5 ilustra uma comparação das trajetórias do veículo relativa a cada treinamento feito.

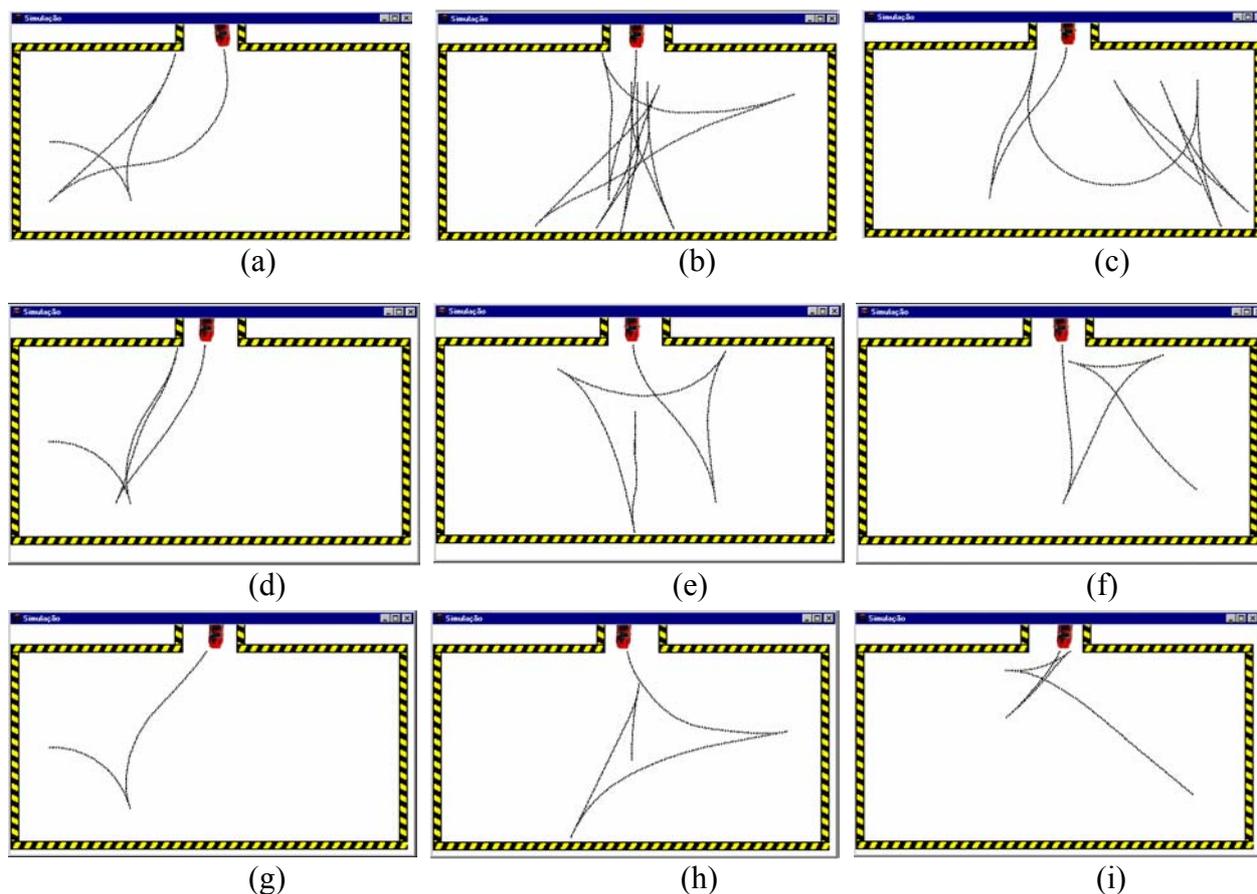


Figura 5.5 – Comparações das trajetórias do treinamento de regras + FP's de GEN1. (a) Posição 1-sem treinamento (b) Posição 2-sem treinamento (c) Posição 3-sem treinamento (d) Posição 1-treinamento de FP's (e) Posição 2-treinamento de FP's (f) Posição 3-treinamento de FP's (g) Posição 1-treinamento de regras+FP's (h) Posição 2-treinamento de regras+FP's (i) Posição 3-treinamento de regras+FP's.

Como mostrado na Tabela 5.5 e ilustrado na Figura 5.5, obteve-se uma redução de 1206 iterações (64,4%) para o veículo estacionar partindo-se das posições iniciais fixadas para o treinamento. O Anexo D mostra o conjunto de regras modificado pelo treinamento das regras mais funções de pertinência (GEN1 RegrasFP). A Figura 5.6 mostra as funções de pertinência originais e a Figura 5.7 mostra as funções de pertinência após o ajuste (ambas foram colocadas juntas para efeito de visualização das modificações nas curvas).

Com relação ao executado em [1], o treinamento de regras juntamente com as funções de pertinência apresentou melhora quanto ao número de iterações, demonstrando a eficácia deste.

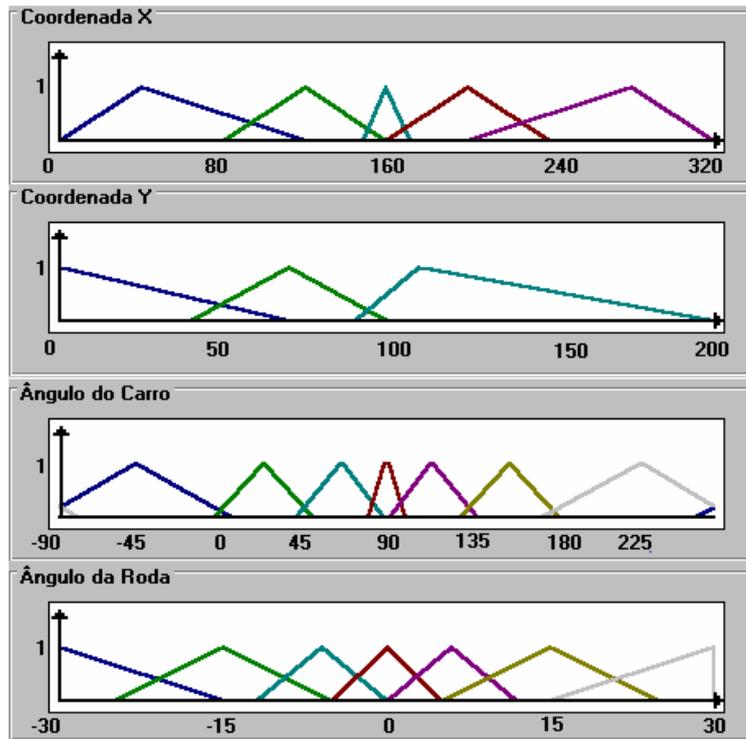


Figura 5.6 – Funções de pertinência originais

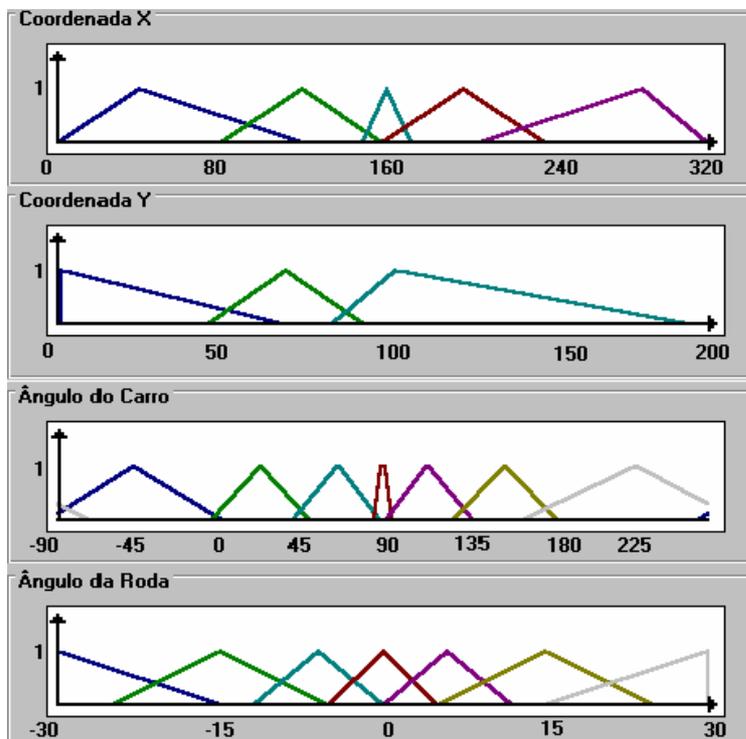


Figura 5.7 – Funções de pertinência após o ajuste de regras + FP's

### 5.3. O controle GEN2

O treinamento deste controle foi feito a partir de sete posições iniciais conforme mostra a Tabela 5.6.

Tabela 5.6 – Posições iniciais para o controle GEN2.

Posição	X	Y	Ângulo do Carro	Iterações sem treinamento
1	5	195	212	Sem solução
2	30	120	250	449
3	80	195	-50	328
4	160	100	-90	873
5	240	194	220	320
6	275	140	-38	746
7	313	194	-56	496

A Figura 5.8 mostra o veículo em cada uma das posições iniciais.

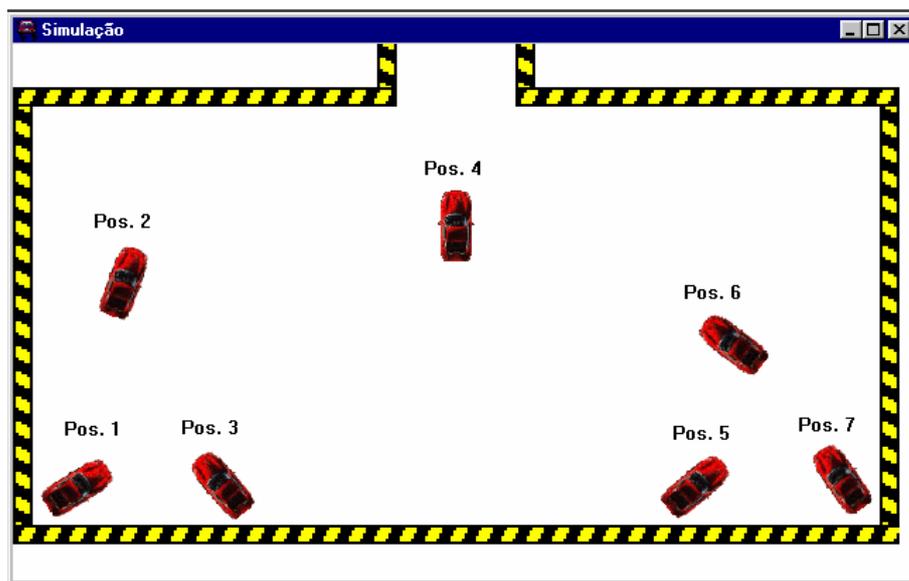


Figura 5.8 – Posições iniciais do treinamento de GEN2.

A Figura 5.9 mostra as trajetórias referentes a cada posição inicial. A Figura 5.8 (a) mostra uma posição sem solução considerada pelo algoritmo genético como uma posição de 1000 iterações.

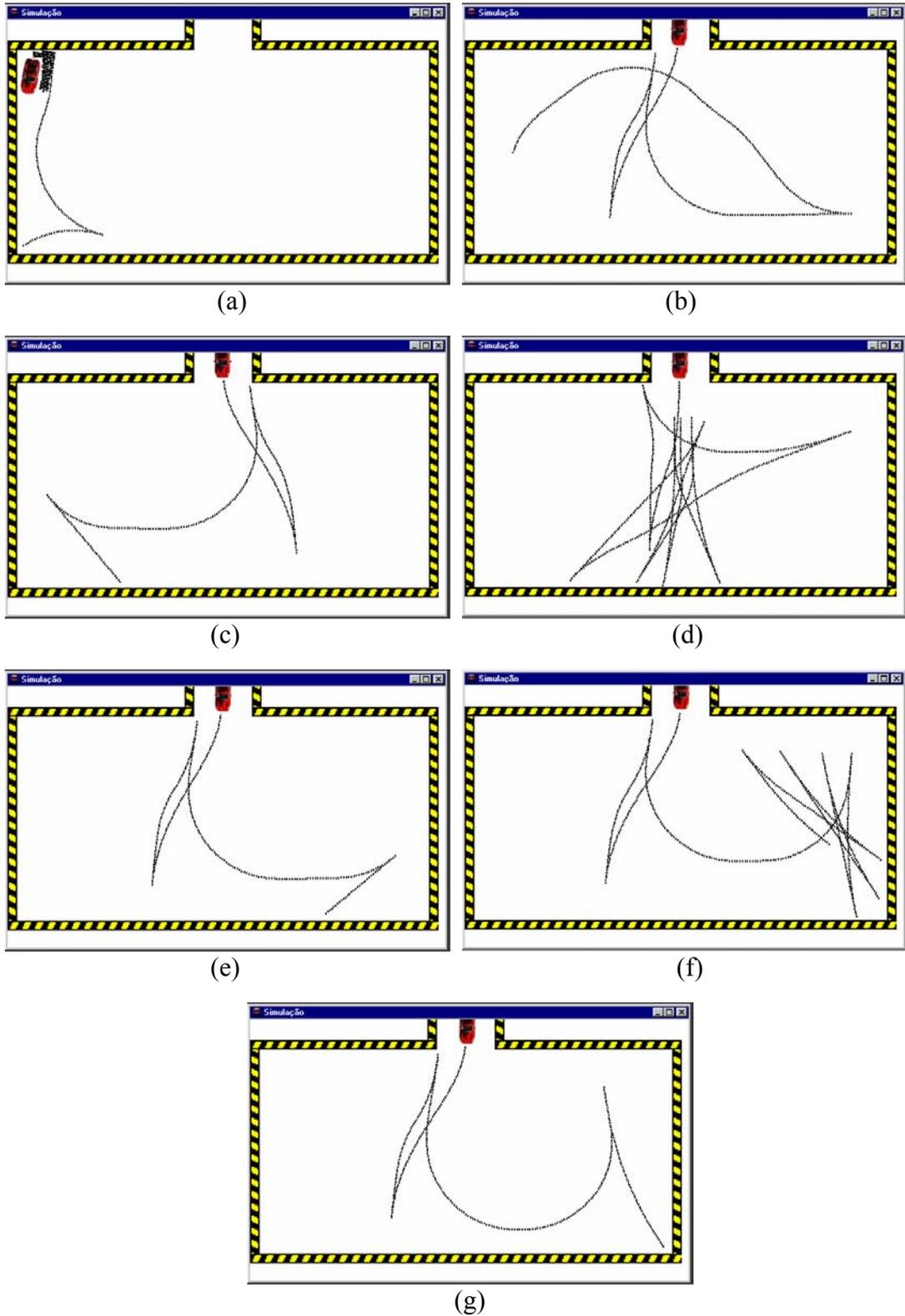


Figura 5.9 - Simulações sem treinamento.  
 (a) Posição 1 – (b) Posição 2 – (c) Posição 3 – (d) Posição 4  
 (e) Posição 5 – (f) Posição 6 – (g) Posição 7

Foram feitos 2 testes diferentes para a verificação da eficiência dos mecanismos utilizados, os quais serão mostrados a seguir.

### 5.3.1. Treinamento de Regras para o controle GEN2

Os parâmetros genéticos definidos para o treinamento são mostrados na Tabela 5.7. Foram escolhidos os mesmos parâmetros de [1] para a comparação dos resultados.

Tabela 5.7 – Parâmetros Genéticos para o controle GEN2.

Tamanho da População	30
Número de Gerações	80
Probabilidade de Crossover	90%
Probabilidade de Mutação	1%

O resultado gerado pelos algoritmos genéticos são mostrados na Tabela 5.8. Foi colocado também o resultado obtido em [1] para efeito de comparação dos resultados.

Tabela 5.8 – Iterações após o treinamento genético do controle GEN2.

Posição	Iterações sem Treinamento	Iterações com treinamento de FP's*	Iterações com treinamento de regras
1	1000 (Sem solução)	252	244
2	449	197	158
3	328	174	172
4	873	396	250
5	320	314	169
6	746	249	194
7	496	348	189
Total	4212	1930	1376
Média	601,71	253,71	196,57

\*Resultados de [1]

A Figura 5.10 ilustra uma comparação das trajetórias do veículo relativa a cada treinamento feito para as sete posições do controle GEN2.

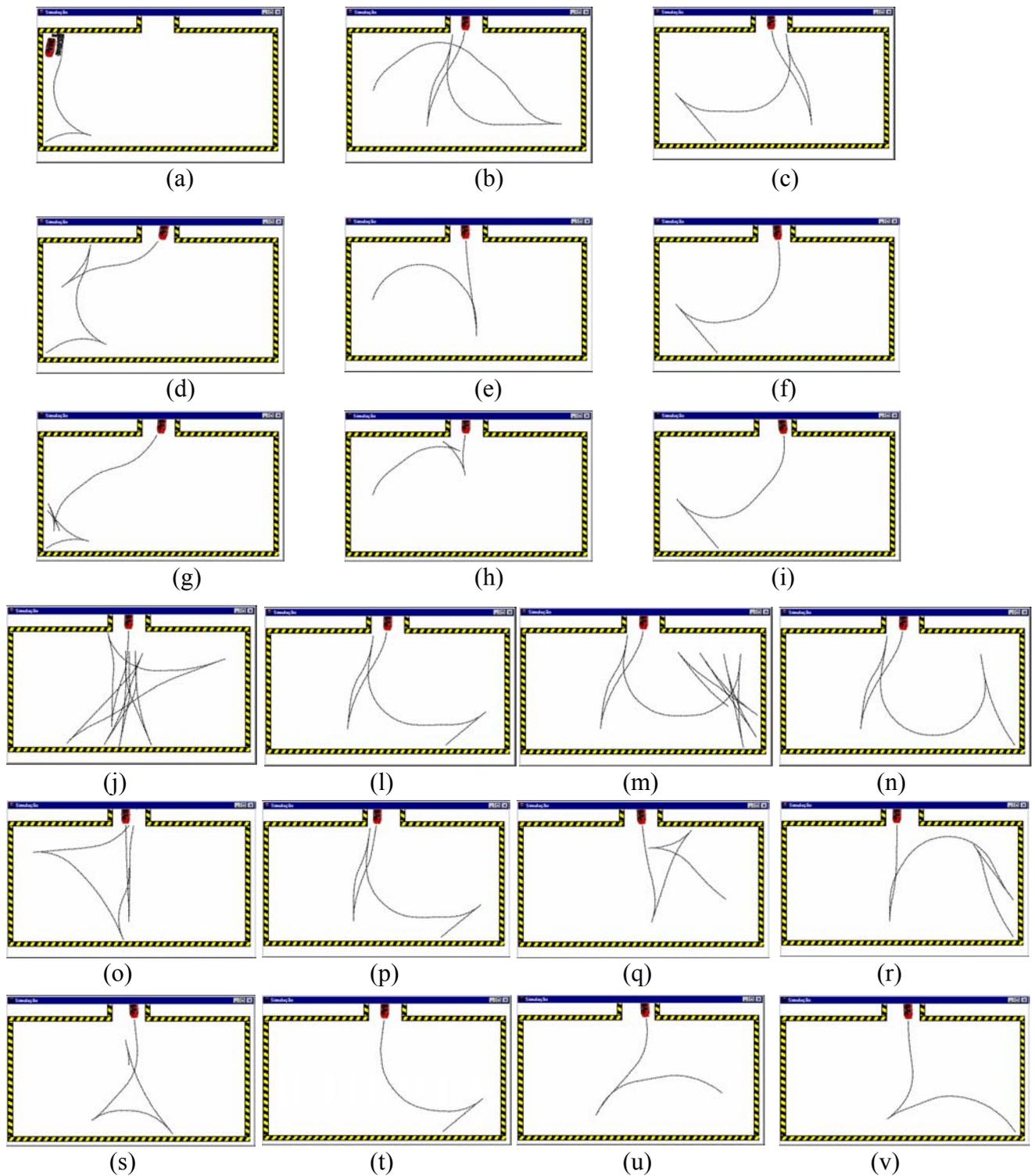


Figura 5.10 – Comparações das trajetórias do treinamento de regras de GEN2. (a) Posição 1-sem treinamento (b) Posição 2-sem treinamento (c) Posição 3-sem treinamento (d) Posição 1-treinamento de FP's (e) Posição 2-treinamento de FP's (f) Posição 3-treinamento de FP's (g) Posição 1-treinamento de regras (h) Posição 2-treinamento de regras (i) Posição 3-treinamento de regras (j) Posição 4-sem treinamento (l) Posição 5-sem treinamento (m) Posição 6-sem treinamento (n) Posição 7-sem treinamento (o) Posição 4-treinamento de FP's (p) Posição 5-treinamento de FP's (q) Posição 6-treinamento de FP's (r) Posição 7-treinamento de FP's (s) Posição 4-treinamento de regras (t) Posição 5-treinamento de regras (u) Posição 6-treinamento de regras (v) Posição 7-treinamento de regras.

Obteve-se com este treinamento uma redução de 2836 iterações (67,3%) para o veículo estacionar partindo-se das posições iniciais fixadas para o treinamento. O Anexo C mostra o conjunto de regras modificado pelo treinamento das regras (GEN2 Regras). Nota-se que o treinamento de regras melhorou a trajetória do veículo, comprovando a eficácia deste.

### 5.3.2. Treinamento de Regras + FP's para o controle GEN2

Os parâmetros genéticos definidos para o treinamento são mostrados na Tabela 5.9. Foram escolhidos os mesmos parâmetros de [1] para a comparação dos resultados.

Tabela 5.9 – Parâmetros Genéticos para o controle GEN2.

Tamanho da População	30
Número de Gerações	80
Probabilidade de Crossover	90%
Probabilidade de Mutação	1%

O resultado gerado pelos algoritmos genéticos é mostrado na Tabela 5.10

Tabela 5.10 – Iterações após o treinamento genético do controle GEN2.

Posição	Iterações sem Treinamento	Iterações com treinamento de FP's*	Iterações com treinamento de regras + FP's
1	1000 (Sem solução)	252	208
2	449	197	158
3	328	174	163
4	873	396	243
5	320	314	163
6	746	249	185
7	496	348	188
Total	4212	1930	1308
Média	601,71	253,71	186,86

\*Resultados de [1]

A Figura 5.11 ilustra uma comparação das trajetórias do veículo relativa a cada treinamento feito para as sete posições do controle GEN2.

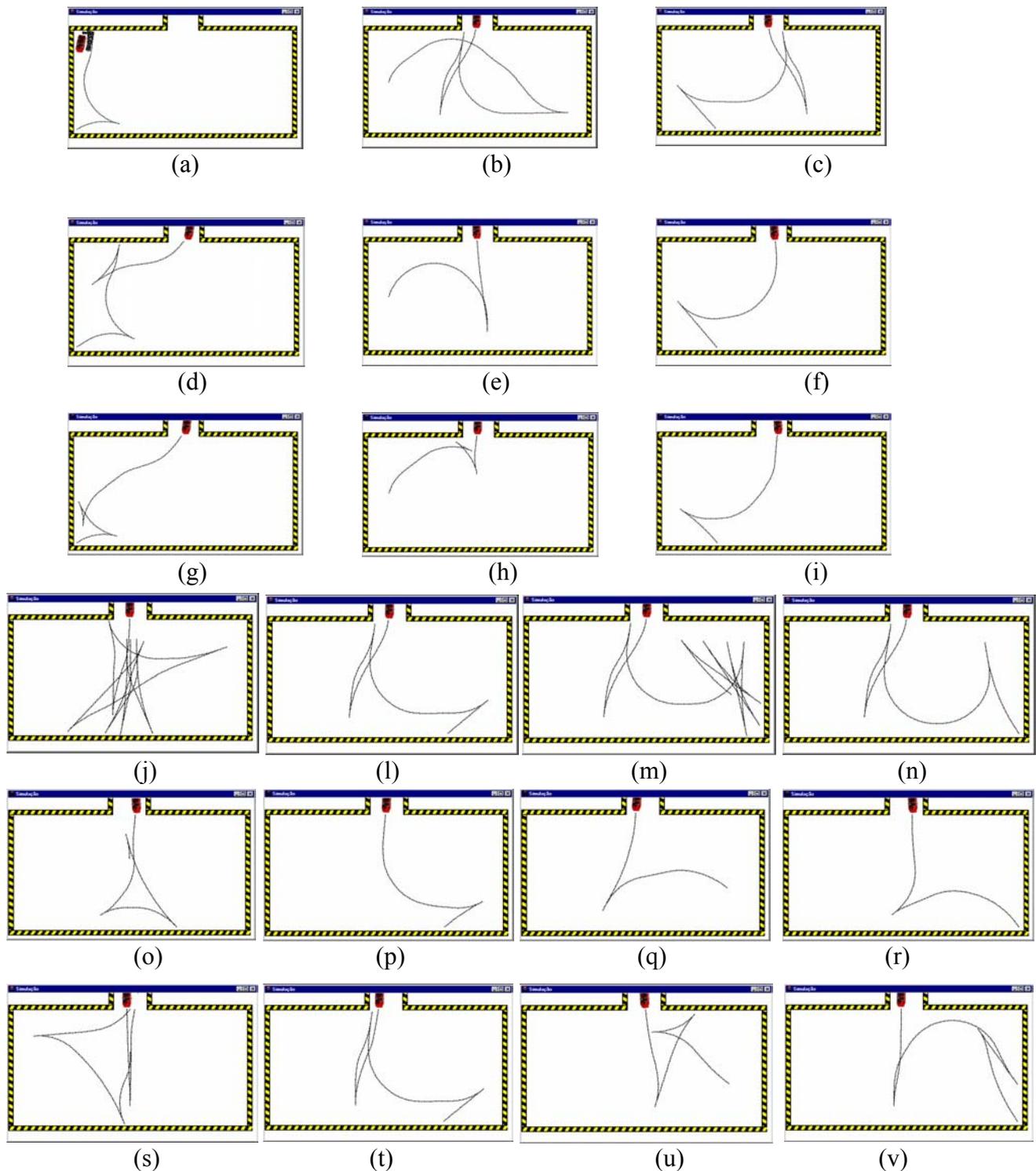


Figura 5.11 – Comparações das trajetórias do treinamento de regras + FP's de GEN2.  
 (a) Posição 1-sem treinamento (b) Posição 2-sem treinamento (c) Posição 3-sem treinamento  
 (d) Posição 1-treinamento de FP's (e) Posição 2-treinamento de FP's (f) Posição 3-treinamento de FP's  
 (g) Posição 1-treinamento de regras+FP's (h) Posição 2-treinamento de regras+FP's (i) Posição 3-treinamento de regras+FP's  
 (j) Posição 4-sem treinamento (l) Posição 5-sem treinamento (m) Posição 6-sem treinamento (n) Posição 7-sem treinamento  
 (o) Posição 4-treinamento de FP's (p) Posição 5-treinamento de FP's (q) Posição 6-treinamento de FP's (r) Posição 7-treinamento de FP's  
 (s) Posição 4-treinamento de regras+FP's (t) Posição 5-treinamento de regras+FP's (u) Posição 6-treinamento de regras+FP's (v) Posição 7-treinamento de regras+FP's.

Obteve-se com este treinamento uma redução de 2904 iterações (68,9%) para o veículo estacionar partindo-se das posições iniciais fixadas para o treinamento. O Anexo E mostra o conjunto de regras modificado pelo treinamento das regras mais funções de pertinência (GEN2 RegrasFP) e a Figura 5.12 mostra as funções de pertinência originais e as funções de pertinência após o ajuste para possibilitar a visualização dos ajustes nas curvas.

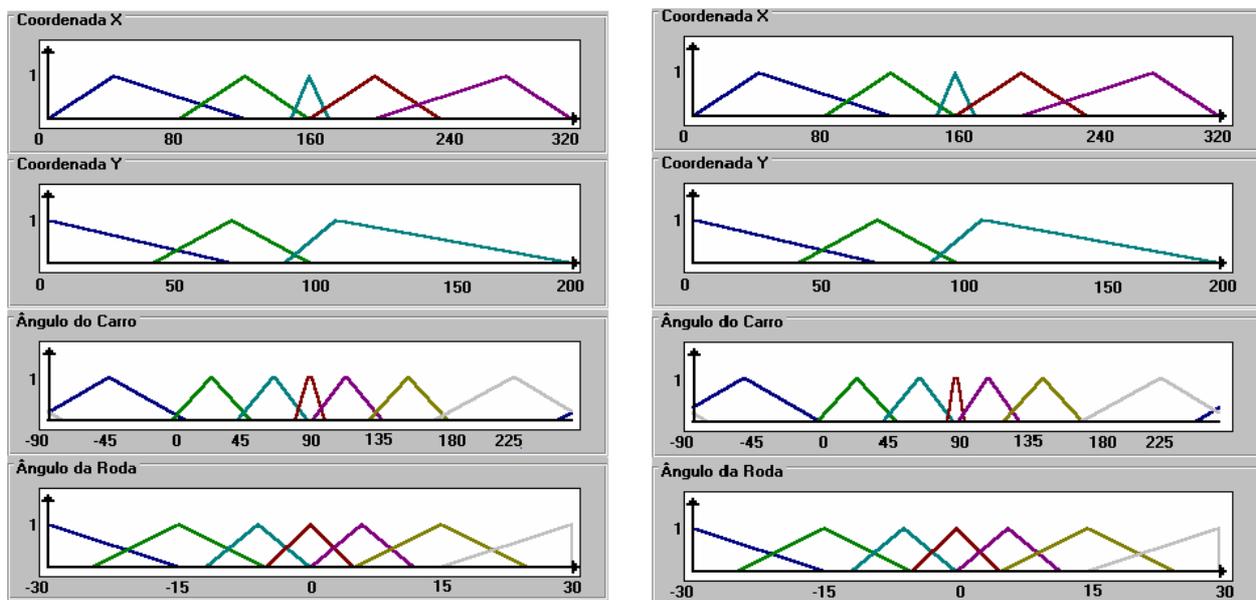


Figura 5.12 – Uma comparação das FP's originais com as FP's após o ajuste.

## 5.4. Resultados de Simulações

A Tabela 5.11 apresenta o resultado obtido de simulações feitas com 30 posições escolhidas aleatoriamente para o veículo estacionar utilizando o controle original RE, que teve suas regras e funções de pertinência ajustadas pelos algoritmos genéticos gerando os controles GEN1 Regras, GEN1 RegrasFP, GEN2 Regras e GEN2 RegrasFP.

Os resultados demonstram uma redução média do número de iterações para o veículo atingir a posição final de 23% para o controle GEN1 Regras, 7% para o controle GEN1 RegrasFP, 2% para o controle GEN2 Regras e 3% para o controle

GEN2 RegrasFP. Estes valores representam uma redução global na trajetória do veículo partindo-se de posições não utilizadas no treinamento genético, mas em comparação com os valores obtidos em [1], o resultado não foi satisfatório (exceto para o controle GEN1).

Tabela 5.11 – Resultados de simulações.

Posição	X	Y	Ângulo do Carro	Iterações geradas pelos Controles Difusos				
				RE	GEN1 Regras	GEN1 RegrasFP	GEN2 Regras	GEN2 RegrasFP
1	1	126	182	451	563	1000	271	284
2	6	46	132	168	382	1000	299	185
3	8	41	190	1000	1000	1000	1000	1000
4	10	187	228	454	297	362	299	299
5	15	70	-90	319	384	408	172	149
6	51	112	48	279	497	1000	249	292
7	51	112	54	281	559	1000	253	297
8	70	95	-40	276	267	269	1000	1000
9	74	69	190	165	214	212	290	388
10	76	193	232	606	240	232	531	322
11	88	46	44	284	1000	1000	1000	1000
12	115	120	0	183	156	218	148	147
13	120	90	45	183	50	109	166	183
14	131	140	-72	458	365	279	1000	1000
15	141	69	-28	343	276	364	1000	1000
16	154	166	-80	1000	371	579	269	1000
17	160	135	268	1000	380	480	401	257
18	161	191	178	1000	208	200	195	191
19	173	140	-72	1000	1000	297	1000	1000
20	208	143	244	364	309	438	1000	340
21	217	66	-50	685	438	513	306	1000
22	228	194	-48	1000	390	596	151	155
23	246	169	154	313	162	156	157	158
24	250	180	-40	740	396	477	164	169
25	265	170	-40	673	238	238	177	182
26	290	95	-40	281	289	434	1000	328
27	300	124	258	318	168	163	160	164
28	305	156	-90	522	342	347	180	177
29	314	73	-46	236	293	309	1000	1000
30	314	194	-44	514	352	350	1000	1000
<b>Total</b>				<b>15096</b>	<b>11586</b>	<b>14030</b>	<b>14838</b>	<b>14667</b>
<b>Média</b>				<b>503,2</b>	<b>386,2</b>	<b>467,67</b>	<b>494,6</b>	<b>488,9</b>

Com relação aos valores da Tabela 5.10 de [1], existem alguns valores diferentes porque foi utilizado um artifício no programa que realiza a trajetória de cada posição automaticamente e soma estes valores com o objetivo de automatizar este cálculo manual. Quando aparece o valor 1000, isso significa que o veículo demorou mais do que 800 iterações para estacionar na garagem ou que ele não conseguiu estacionar na garagem.

Observando-se os valores obtidos na Tabela 5.11, conclui-se que:

- a) Existem posições que não conseguem ser melhoradas, como a posição 3 que permanece com o número de iterações igual a 1000 por ser uma posição de difícil manobra.
- b) Em determinadas posições o número de iterações é maior que as geradas pelos controles originais (sem treinamento). Na posição 11, o controle RE gera 284 iterações para estacionar o veículo enquanto que todos os outros controles geram 1000 iterações. Este aumento é em consequência das modificações nas regras e nas funções da pertinência que fazem com que o veículo desenvolva uma trajetória diferente para atingir a posição final.
- c) Um valor menor de número de iterações obtido do treinamento para algumas posições não implica na redução global de várias posições. Estes valores estão relacionados às posições iniciais utilizadas nas simulações e eles variam quando se utilizam posições iniciais diferentes. Mas verificou-se que sempre há uma redução do número de iterações.

## Capítulo 6

### Conclusão e Trabalhos Futuros

O treinamento de regras mais funções de pertinência é sempre melhor do que só o treinamento das funções de pertinência ou só o treinamento das regras. Isso porque é como se o treinamento de regras mais funções de pertinência rodasse o treinamento das funções de pertinência diversas vezes, como se fossem muitas gerações dentro de um só programa, com o adicional de procura de melhores regras também. Pensou-se nesta junção do treinamento de regras juntamente com o treinamento das funções de pertinência, pois se os dois melhoram o controle original a união dos dois melhoraria muito mais as regras e as funções de pertinência. Os resultados comprovam esta melhora.

Foi visto que como o treinamento das regras utiliza um procedimento condicional devido a necessidade de não perder-se a lógica das regras, os valores obtidos de cada cromossomo são parecidos desde o início. Enquanto que, pelo fato do treinamento das funções de pertinência utilizar um procedimento aleatório (pois as curvas são representadas por números em que se pode somar ou subtrair valores), os valores são bem distintos e, de repente, ocorre um ótimo valor que será o melhor cromossomo.

Com a utilização da técnica de otimização Algoritmos Genéticos aplicada ao problema de controle difuso foi possível a obtenção de melhores resultados, comprovando sua eficácia. Esta técnica mostra que o desempenho de um controle difuso pode ser melhorado através dos algoritmos genéticos.

Com o acréscimo de mais posições a serem treinadas, o programa se tornou lento devido ao treinamento de várias gerações de funções de pertinência para cada posição do carrinho (conforme descrito no Capítulo 3, item 3.4). Mas essa lentidão foi contornada fazendo com que o programa rodasse menos gerações a cada vez e com o resultado obtido dessas gerações, fosse inicializado novamente o programa até que alcançasse o número de gerações pretendido para os testes. Como sugere Schwefel,

um notável pesquisador da computação evolucionária: “O foco deveria ser no melhoramento ao invés da otimização”. Com relação ao treinamento de várias posições ao mesmo tempo, o programa foi melhorado (pois reduziu o tempo gasto para o carrinho entrar na garagem), não otimizado (devido à sua lentidão). Mas, com a utilização do artifício de execução do programa com menos gerações foram obtidos resultados satisfatórios comprovando a eficiência do programa. O que se procura no mundo real é satisfação com soluções eficientes.

Como trabalhos futuros e complementares a este pacote computacional pode-se sugerir:

- a) a utilização de outros métodos de otimização para efeito de comparação com os Algoritmos Genéticos, como por exemplo, a técnica de Particle Swarm Optimization;
- b) um método que fuzzificasse a área próxima à garagem para que qualquer regra utilizada pelo usuário fosse modificada nessa área com o objetivo do carrinho entrar mais rapidamente na garagem;
- c) fazer a inserção e remoção de funções de pertinência através dos algoritmos genéticos;
- d) a inclusão de uma lógica que determine se a melhor saída para o veículo é mover-se para frente ou para trás (atualmente, ele sai sempre para frente);
- e) o desenvolvimento de barreiras na área do estacionamento, gerando dificuldades ao sistema de controle.

## Referências Bibliográficas

- [1] M. A. Carvalho, *Ajuste de Funções de Pertinência utilizando Algoritmos Genéticos*, Dissertação de Mestrado, Escola Federal de Engenharia de Itajubá , 2000.
- [2] S. G. de Brito, *Pacote Computacional para o Ensino da Lógica Difusa*, Dissertação de Mestrado, Escola Federal de Engenharia de Itajubá , 1998.
- [3] L. A. Zadeh, “Outline of a New Approach to the Analysis of Complex Systems and Decision Processes,” *IEEE Trans. Systems, Man, and Cybernetics*, Vol. SMC-3, 1973, pp. 28-44.
- [4] D. Dubois & H. Prade – *Fuzzy Sets and Systems: Theory and Applications*, New York: Academic Press, 1980.
- [5] J. Pearl – *Heuristics – Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley Pub. Co., 1985.
- [6] J. Kennedy, R. C. Eberhart – *Swarm Intelligence*, Morgan Kaufmann, 2001.
- [7] J. H. Holland, *Adaptation in natural and artificial systems* (The University of Michigan Press, Ann Arbor, 1975), reprinted by MIT Press, 1992.
- [8] David E. Goldberg, *Genetic algorithms in search, optimization and machine learning*, Addison Wesley, 1989.
- [9] A. Laprade & G. Lambert-Torres - "Controlando um Veículo Utilizando a Teoria dos Conjuntos Nebulosos", Anais I Cong. Bras. Automação Inteligente, pp.165-174, Rio Claro, Set. 1993.
- [10] M. A. Carvalho, G. Lambert-Torres, L. E. Borges da Silva & J. O. P. Pinto, “Fitting Fuzzy Membership Functions using Genetic Algorithms”, in *2000 IEEE International Conference on Systems, Man, and Cybernetics*, Oct. 8-11 2000, Nashville, USA.
- [11] Y. J. Túpac, M. A. Pacheco, M. Vellasco, R. Tanscheit, “Geração do Conjunto de Regras de Inferência para um Controlador Nebuloso usando Algoritmos Genéticos”, *4<sup>o</sup> SBAI – Simpósio Brasileiro de Automação Inteligente*, 8-10 Setembro 1999, São Paulo.
- [12] E. H. Mamdani, “Application for fuzzy algorithms for the control of a dynamic plant,” *Proc. IEEE.* , vol. 121, pp. 1585-1588, 1974.
- [13] D. Park & A. Kandel, - “Genetic-Based New Fuzzy Reasoning Models with Application to Fuzzy Control”, *IEEE Trans. On Systems*, vol. 24, no. 1, pp. 39-47,

January 1994.

- [14] L.R. Medsker - *Hybrid Intelligent Systems*, Boston: Kulwer Academic Pub., 1995.
- [15] D. Driankov, H. Hellendoorn, M. Reinfrank – *An Introduction to Fuzzy Control*, Springer-Verlag, 1995.
- [16] J. B. Kiszka, M. E. Kochanska, and D. S. Sliwiska, “The influence of some fuzzy implication operators on the accuracy of a fuzzy model, Part I,” *Fuzzy Sets and Syst.*, vol. 15, pp. 111-128, 1985.
- [17] \_\_\_\_\_, “The influence of some fuzzy implication operators on the accuracy of a fuzzy model, Part II,” *Fuzzy Sets and Syst.*, vol. 15, pp. 223-240, 1985.

## Anexos

### Anexo A

#### Estacionamento Difuso – Regras: Regras Originais

Se X =	LE	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YM	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YM	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	,	Y =	YM	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	LE	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LE	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LC	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YT	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YM	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	LC	,	Y =	YM	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YM	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LC	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	RC	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	RC	,	Y =	YT	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	RC	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	NS





## Anexo B

### Estacionamento Difuso – Regras: GEN1 Regras

Se X =	LE	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YT	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YT	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YM	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LE	, Y =	YM	, Ângulo =	RU	, Dir. =	Frente	Então	Inverte Direção	
Se X =	LE	, Y =	YM	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	, Y =	YM	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	, Y =	YM	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	LE	, Y =	YM	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YM	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	, Y =	YB	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	, Y =	YB	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YB	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	, Y =	YB	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LC	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YT	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LC	, Y =	YT	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YT	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YT	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	LC	, Y =	YM	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YM	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YM	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YM	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YM	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YB	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LC	, Y =	YB	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	, Y =	YB	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	, Y =	YB	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YT	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	, Y =	YT	, Ângulo =	LB	, Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	, Y =	YM	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YM	, Ângulo =	LU	, Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	, Y =	YB	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	, Y =	YB	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	, Y =	YB	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YB	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	RC	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	RC	, Y =	YT	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	RC	, Y =	YT	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	NS





Se X = RE , Y = YB , Ângulo = VE , Dir. = Trás Então Ângulo Roda = PB  
Se X = RE , Y = YB , Ângulo = LV , Dir. = Trás Então Ângulo Roda = PB  
Se X = RE , Y = YB , Ângulo = LU , Dir. = Trás Então Ângulo Roda = PS  
Se X = RE , Y = YB , Ângulo = LB , Dir. = Trás Então Ângulo Roda = NB

**Listadas : 162 regras.**

## Anexo C

### Estacionamento Difuso – Regras: GEN2 Regras

Se X =	LE	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	,	Y =	YT	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YM	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LE	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YM	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LE	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LC	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LC	,	Y =	YT	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LC	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YM	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YM	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LC	,	Y =	YM	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YB	,	Ângulo =	LV	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	LC	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	,	Y =	YM	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	,	Y =	YB	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	RC	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NM





Se X = RE , Y = YB , Ângulo = LV , Dir. = Trás Então Ângulo Roda = NB  
Se X = RE , Y = YB , Ângulo = LU , Dir. = Trás Então Ângulo Roda = ZE  
Se X = RE , Y = YB , Ângulo = LB , Dir. = Trás Então Ângulo Roda = NB

**Listadas : 161 regras.**

## Anexo D

### Estacionamento Difuso – Regras: GEN1 Regras FP

Se X =	LE	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YM	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	LE	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	LE	,	Y =	YM	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	,	Y =	YM	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	,	Y =	YM	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	LE	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YM	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YT	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LC	,	Y =	YT	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	LC	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YM	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YM	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YM	,	Ângulo =	LV	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LC	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	CE	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YT	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	,	Y =	YT	,	Ângulo =	LB	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YM	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YM	,	Ângulo =	LU	,	Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	,	Y =	YB	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	ZE
Se X =	CE	,	Y =	YB	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	,	Y =	YB	,	Ângulo =	LU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	RC	,	Y =	YT	,	Ângulo =	RB	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	,	Y =	YT	,	Ângulo =	RU	,	Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	,	Y =	YT	,	Ângulo =	RV	,	Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	RC	,	Y =	YT	,	Ângulo =	VE	,	Dir. =	Frente	Então	Ângulo Roda =	PS





Se X = RE , Y = YB , Ângulo = RV , Dir. = Trás Então Ângulo Roda = PS  
Se X = RE , Y = YB , Ângulo = VE , Dir. = Trás Então Ângulo Roda = PB  
Se X = RE , Y = YB , Ângulo = LV , Dir. = Trás Então Ângulo Roda = PB  
Se X = RE , Y = YB , Ângulo = LU , Dir. = Trás Então Ângulo Roda = PS  
Se X = RE , Y = YB , Ângulo = LB , Dir. = Trás Então Ângulo Roda = NB

**Listadas : 163 regras.**

## Anexo E

### Estacionamento Difuso – Regras: GEN2 Regras FP

Se X =	LE	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	, Y =	YT	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YT	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YT	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YT	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YM	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	, Y =	YM	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LE	, Y =	YM	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LE	, Y =	YM	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LE	, Y =	YB	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LE	, Y =	YB	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	, Y =	YT	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YT	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	LC	, Y =	YT	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YT	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YM	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YM	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	LC	, Y =	YM	, Ângulo =	LV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YM	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	LC	, Y =	YB	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	, Y =	YB	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	LC	, Y =	YB	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	LC	, Y =	YB	, Ângulo =	LV	, Dir. =	Frente	Então	Inverte Direção	
Se X =	LC	, Y =	YB	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	CE	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	, Y =	YT	, Ângulo =	LU	, Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	, Y =	YT	, Ângulo =	LB	, Dir. =	Frente	Então	Inverte Direção	
Se X =	CE	, Y =	YM	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	PS
Se X =	CE	, Y =	YM	, Ângulo =	VE	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	, Y =	YM	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	PB
Se X =	CE	, Y =	YM	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YB	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NS
Se X =	CE	, Y =	YB	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	PM
Se X =	CE	, Y =	YB	, Ângulo =	LU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	CE	, Y =	YB	, Ângulo =	LB	, Dir. =	Frente	Então	Ângulo Roda =	NM
Se X =	RC	, Y =	YT	, Ângulo =	RB	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	, Y =	YT	, Ângulo =	RU	, Dir. =	Frente	Então	Ângulo Roda =	NB
Se X =	RC	, Y =	YT	, Ângulo =	RV	, Dir. =	Frente	Então	Ângulo Roda =	NM





Se X =	RE	,	Y =	YB	,	Ângulo =	VE	,	Dir. =	Trás	Então	Ângulo Roda =	PB
Se X =	RE	,	Y =	YB	,	Ângulo =	LV	,	Dir. =	Trás	Então	Ângulo Roda =	NB
Se X =	RE	,	Y =	YB	,	Ângulo =	LU	,	Dir. =	Trás	Então	Ângulo Roda =	PS
Se X =	RE	,	Y =	YB	,	Ângulo =	LB	,	Dir. =	Trás	Então	Ângulo Roda =	NB

**Listadas: 162 regras.**