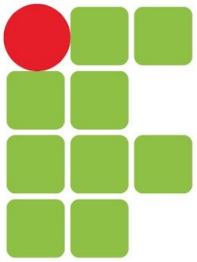


INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SÃO PAULO
Câmpus Guarulhos

Relatório final de iniciação científica

Campus Guarulhos

Estudo de caso: Construção de um robô autónomo na plataforma arduino controlado por lógica fuzzy

Aluno:José Paulo Junior Alves da Silva
Orientador:Percy Javier Igei Kaneshiro

Junho de 2018

RESUMO

Este projeto de iniciação científica é parte de um projeto maior de construção de robôs autônomos do grupo de estudos em robótica e sistemas embarcados (GERSE) do IFSP-Guarulhos. Neste projeto, o objetivo é estudar a aplicação da lógica fuzzy no desenvolvimento de um robô autônomo utilizando a plataforma ARDUINO. O sistema de controle do robô autônomo deve ser projetado de tal forma que possibilite uma correta execução das atividades que lhe são atribuídas, mesmo considerando a presença de possíveis imprecisões, erros ou eventos externos aleatórios. Neste contexto, este trabalho consiste na aplicação de um procedimento para modelar e simular as estratégias de controle dos robôs móveis. Este procedimento utiliza a rede de Petri como linguagem de modelagem e a lógica fuzzy para controlar as funcionalidades do robô autônomo. Para ilustrar as principais características desta abordagem, será modelado e implementado um robô móvel seguidor de linha, utilizando a plataforma ARDUINO. A partir dos modelos desenvolvidos será possível fazer uma análise do comportamento do sistema, identificando condições indesejáveis, como a presença de estados onde o sistema fica travado ou a detecção de erros de especificação, prevenindo problemas em fases posteriores ao projeto.

Palavras-chaves: Rede de Petri, Lógica Fuzzy, Sistemas fuzzy, Inteligencia artificial, Robótica móvel e Arduino.

ABSTRACT:

This project derived from the scientific initiation is part of a larger project of construction of autonomous robots of the group of studies in robotics and embedded systems (GERSE) of IFSP-Guarulhos. In this project, the objective is to study the application of fuzzy logic in the development of an autonomous robot using the ARDUINO platform. The control system of the autonomous robot was designed in such a way that it enabled a correct execution of the activities attributed to it, even considering the presence of possible inaccuracies, errors or random external events. In this context, this work consists in the application of a procedure to model and simulate the robot control strategies. This procedure uses the Petri net as a modeling language and the fuzzy logic to control the functionalities of the autonomous robot. To illustrate the main characteristics of this approach, a mobile robotic line follower was modeled and implemented using the ARDUINO platform. From the developed models it was possible to make an analysis of the behavior of the system, identifying undesirable conditions, such as the presence of states where the system is locked or detection of specification errors, preventing problems in later phases of the project.

KEYWORDS: Petri Net, Fuzzy Logic, Fuzzy Systems, Artificial Intelligence, Mobile Robotics and Arduino.

SUMÁRIO

1 INTRODUÇÃO

Este projeto de iniciação científica é parte de um projeto maior de construção de robôs autónomos do grupo de estudos em robótica e sistemas embarcados (GERSE) do IFSP Guarulhos. Em 2017 e 2018 tivemos a participação na olimpíada brasileira de robótica (OBR), e no torneio de robótica dos institutos federais (TRIF), onde o objetivo era fazer o resgate de uma vítima por meio de um robô autónomo. Foi Utilizada a plataforma LEGO MINDSTORMS Education EV3 que é uma solução educacional de robótica, que estimula o Aprendizado de STEM (sigla internacional para as áreas de Ciências, Tecnologia, Engenharia e Matemática). Esse foi o ponto de partida para compreender a importância da robótica em impactar algum projeto e sua utilidade no cotidiano civil/militar.
A iniciação científica é uma continuação do trabalho de construção de robôs autônomos, porém o estudo de caso terá componentes diferentes dos que foram utilizados. Neste estudo de caso, está sendo utilizada uma plataforma Arduino que é uma plataforma de prototipagem eletrônica de hardware livre e de placa única. Esta plataforma utiliza um microcontrolador Atmel AVR com suporte de entrada e saída embutido e uma linguagem de programação padrão. Neste caso o chassi do robô autônomo será criado por material convencional diferentemente do uso das peças disponíveis pelo kit lego Ev3.

2 REVISÃO DA LITERATURA

A iniciação científica terá como diretriz o estudo de caso sobre um robô autônomo controlado por lógica fuzzy. O objetivo deste projeto consiste em aplicar um método para especificar as funcionalidades do sistema de controle de um robô autônomo. A modelagem das funcionalidades será realizada por meio das redes de Petri. A lógica de controle será realizada utilizando a lógica fuzzy e todos seus processos que são diferentes frente à lógica booleana. Finalmente, o robô autônomo e a lógica de controle serão utilizados em um kit de robótica ARDUINO. Atualmente, dentro das linhas de pesquisa do IFSP não existem projetos relacionados à modelagem e especificação de robôs móveis utilizando uma abordagem de sistemas a eventos discretos e as redes de Petri (Miyagi, et al., 2002). Neste sentido, o desenvolvimento deste projeto é uma contribuição fundamental nas pesquisas relacionadas à robótica móvel no IFSP. Neste projeto, o objetivo é estudar a aplicação da lógica fuzzy (RIZOL,2011) no desenvolvimento de um robô autônomo utilizando a plataforma ARDUINO (GIBB, 2010)(XU, et al., 2015). Com isso adquirir experiência na programação e construção de robôs autônomos utilizando a plataforma proposta para participar de torneios de competição de robótica, sendo assim possível futuramente impactar indivíduos a participarem da robótica.
Para realizar esta pesquisa, foi utilizado microcomputador para desenvolver as atividades de modelagem e análise do algoritmo de controle. A construção do robô e a implementação do algoritmo de controle foram executados na plataforma Arduino (nano).
A robótica é uma área de conhecimento que tem evoluído de forma muito rápida nos últimos anos, entretanto o estudo e o projeto de robôs e autômatos vêm sendo desenvolvido há vários séculos. Os primeiros robôs evoluíram de autômatos complexos passando por robôs

manipuladores de base fixa, pelos dispositivos móveis guiados à distância, chegando mais recentemente aos robôs móveis semiautônomos e os totalmente autônomos (JUNG et al., 2005).

2.1 Lógica Fuzzy

O filósofo grego Aristóteles (384-322 a.C.),foi o fundador do estudo da lógica, sendo assim estabeleceu conjuntos de regras que por sua vez eram rígidas para concluir logicamente resultados admissíveis. Ele foi capaz de identificar o problema do ´´meio excluído´´ onde os objetos que podem ser classificados como nem uma coisa nem outra(morno, por exemplo).Logo os matemáticos não tinham nenhum recurso para lidar com valores imprecisos, no século 20 foi apresentado paradoxos.
Em 1920, o lógico Jan Lukasiewicz formulou princípios da lógica multivalor, na qual as afirmações assumem um valor fracionário de verdade entre 1 (verdadeiro) e 0 (falso).
Em 1937, o filósofo Max Black aplicou a lógica multivalor(difusa,ou nebulosa) a conjuntos de objetos e gerou a primeira série de curvas indistintas.
O matemático americano Lofti Zadeh desenvolveu o conceito de lógica indefinida (fuzzy logic) e conjuntos indefinidos (fuzzy sets) em 1965.
A consistência de um valor em um conjunto pode ter um valor no intervalo de [0,1].
Conceito de dualidade que pode afirmar que algo pode e deve coexistir com o seu oposto.
A lógica fuzzy é uma ferramenta capaz de assumir valores vagos e transformam para um formato numérico que são de manipulação aos computadores. Também podemos definir a lógica fuzzy como sendo a lógica que suporta os modos de raciocínio que são aproximados, ao invés de exatos. A lógica em questão combina lógica multivalorada, teoria probabilística,inteligência artificial e redes neurais para que se possa representar o pensamento humano. A restrição que a lógica fuzzy dá aos valores totais dos conjuntos devem resultar em 1 (como por exemplo, 0,2 de frio e 0,6 quente).
Um exemplo de sua aplicação Metrô de Sendai, a Hitachi em 1988 produziu um sistema da lógica fuzzy para operar os trens do metrô em Sendai, no Japão. Os trens precisam de apenas um condutor, e não de um motorista. Os sistemas criados controlam a aceleração, velocidade de cruzeiro e frenagem, levando em conta a segurança,conforto, eficiência do combustível e necessidade de parar exatamente nas posições determinadas as plataformas das estações para embarque.

2.2 Modelagem Fuzzy

Processos de modelagem fuzzy seguem os seguintes passos Fuzzyficar, Inferir, Defuzzificar.
A fuzzificação é usada para converter um valor numérico em um conjunto nebuloso. Esta conversão é feita usando-se as funções de pertinência. O conjunto fuzzy, que reflete a lógica fuzzy, é formado por pares onde um elemento do par representa a variável em estudo

e o outro elemento uma função cuja imagem está contida no intervalo (0,1) e que caracteriza o grau de pertinência da variável. O sistema de inferência fuzzy tem como entrada valores precisos, não fuzzy, que são mapeados pelas funções de pertinência estabelecidas na construção do sistema: Essa é a etapa de fuzzificação. Essa etapa fornece parâmetros fuzzy a uma máquina de inferência que processa uma série de regras do tipo SE-ENTÃO, constituída de proposições, envolvendo termos de variáveis linguísticas. Ao final do processamento das regras, o valor fuzzy, que foi obtido como resposta, é defuzzificado se a máquina de inferência tiver como saída valores fuzzy, obtendo-se assim uma saída precisa novamente. Se a máquina de inferência tiver como saída valores precisos, esses já serão a resposta do sistema chamados de crispi.
O conjunto nebuloso de saída é em geral o resultado da união de uma série de funções de pertinência. A defuzzificação transforma o resultado obtido em valores clássicos. Os métodos de defuzzificação mais utilizados são o centróide, bissetor, entre outros.
O raciocínio fuzzy é um raciocínio que, baseado em dados imprecisos, os quais são representados por graus de pertinência a um conjunto fuzzy, chega a uma conclusão sobre um fenômeno em estudo. O conhecimento do fenômeno é expresso através de afirmativas do tipo: “se (um conjunto de condições é satisfeito) então (pode-se inferir um conjunto de consequências)”.
Um sistema de inferência nebuloso pode ter entradas determinísticas ou fuzzy, mas as saídas sempre serão conjuntos fuzzy. Quando se deseja obter uma saída exata, usa-se um dos métodos de defuzzificação para conseguir o valor que melhor represente o conjunto nebuloso de saída.
A formulação das regras “se… então”, para a criação do sistema de inferência fuzzy é, na prática, bastante difícil. As regras são decididas a priori, baseadas no conhecimento do sistema em estudo.

O raciocínio nebuloso é formado pelos seguintes eventos:

1. Transformação de variáveis numéricas em variáveis nebulosas usando o processo de fuzzificação;
2. Criação de uma base de regras nebulosas expressas por declarações do tipo se. . . Então;
3. Criação de um sistema de inferência que mapeie conjuntos nebulosos em conjuntos fuzzy.

Este sistema manipula como as regras serão combinadas, obtendo-se o resultado utilizando-se um defuzzificador que mapeia o conjunto nebuloso de saída a um número real.
Os sistemas de inferência fuzzy se diferenciam principalmente nos consequentes das suas regras de inferência, e em como é feita a agregação e defuzzificação das saídas.
Um sistema de inferência nebuloso bastante utilizado é o de Mamdani, que tem um conjunto fuzzy como resultado, sendo necessário um procedimento de defuzzificação para se obter um resultado exato. Neste projeto foi utilizado Mandani para calcular os valores crispi.

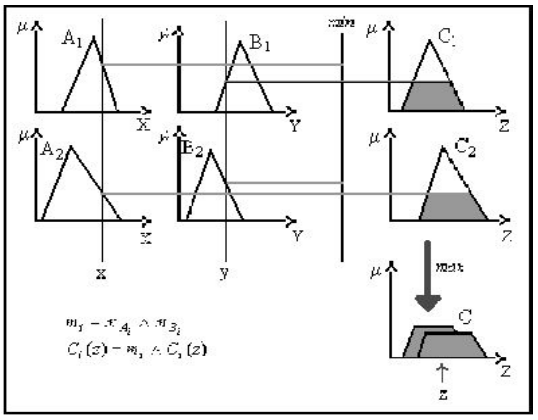


Figura 1:método mamdani exp:centróide

2.3 Redes de Petri

A rede de Petri (PN) foi inicialmente desenvolvido em 1962 por Carl Adam Petri em sua tese de doutorado na Universidade Kommunication mit Automaten de Darmsdadt na antiga Alemanha Ocidental (JENSEN, et al., 2009). É uma ferramenta gráfica e matemática para modelagem, análise e sistemas com paralela, concorrente, assíncrona e processos não-determinístico,especialmente aquelas caracterizadas por estados discretos que mudam abruptamente por eventos assumidos para ser instantânea projetar. O PN permite modelar situações típicas de sistemas distribuídos, como a sincronia, a simultaneidade, a causalidade, os processos de conflito e compartilhamento de recursos.
A qualquer momento durante a execução de uma rede de Petri, cada posição pode armazenar um ou mais tokens. Diferente de sistemas mais tradicionais de processamento de dados, que podem processar somente um único fluxo de tokens entrantes, as transições de redes de Petri podem consumir e mostrar tokens de múltiplos lugares. Uma transição só pode agir nos tokens se o número requisitado de tokens aparecer em cada posição de entrada.
Transições agem em tokens de entrada por um processo denominado disparo. Quando uma transição é disparada, ela consome os tokens de suas posições de entrada, realiza alguma tarefa de processamento, e realoca um número específico de tokens nas suas posições de saída. Isso é feito atomicamente. Como disparos são não determinísticos, redes de Petri são muito utilizadas para modelar comportamento concorrente em sistemas distribuídos.
Uma rede de Petri consiste em *posições*, *transições* e *arcos direcionados*. Arcos interligam posições e transições, não podendo conectar posições e posições ou transições e transições. As posições de entrada de uma transição são aquelas as quais um arco se destina. As posições de saída são aquelas das quais um arco se origina.
Posições podem conter qualquer número de tokens. Transições podem ser disparadas, isto é, executadas: quando uma transição é disparada, ela consome um token de cada uma das suas posições de entrada, e produz um token em cada uma das suas posições de saída. Uma transição é habilitada quando ela pode ser disparada, isto é, quando existem tokens em cada posição de entrada.

A execução de uma rede de Petri é não-determinística. Isso significa que múltiplas transições podem ser habilitadas ao mesmo tempo (cada uma pode ser disparada) e que nenhuma transição deve ser obrigatoriamente executada em determinado momento.

3 MATERIAIS E MÉTODOS

O controlador Arduino é uma plataforma de prototipagem eletrônica de hardware livre e de placa única, projetada com um microcontrolador Atmel AVR com suporte de entrada/saída embutido, uma linguagem de programação padrão, a qual tem origem em Wiring, e é essencialmente C/C++. O objetivo do projeto é criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de se usar por principiantes e profissionais.
Utilizou-se a plataforma Arduíno no robô. Equipado com um microcontrolador Atmega328 (pré-carregado com "bootloader"), possui 14 entradas/saídas digitais (6 podem ser utilizadas como saídas PWN), 6 entradas analógicas, cristal oscilador de 16 MHz, conexão USB, entrada para fonte e conexão USB. O uso do Arduíno é interessante devido ao fato dessa plataforma ser "open-source", ainda é fácil programar com ambiente de desenvolvimento que utiliza a linguagem C a IDE, sendo assim é possível evitar preocupações com a confecção e manutenção de placas, por exemplo criar um modelo em placa de fenolite para fazer um controlador desse porte.
O robô é composto por uma carcaça a qual estão conectados 2 motores, um à esquerda e um a direita. Também estão conectados uma bateria, um circuito driver para os motores, um controlador Arduino nano e um conjunto de sensores óticos(de cores).

Figura 2: Arduino nano

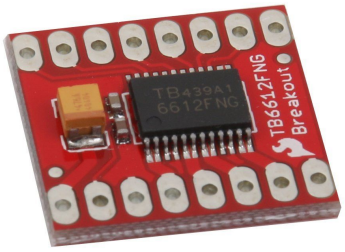


Figura 3:Ponte H modelo Tb6612fng

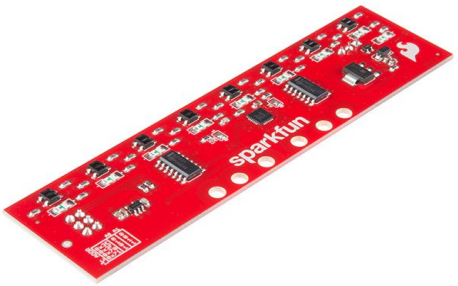


Figura 4: Sensor modelo Line Follower Array

Figura 5: Modelo Motor

Figura 5: Modelo roda

Esses materiais apresentam ótima qualidade e resistência para competições etc. São de baixo custo considerando o papel de desenvolver suas respectivas atividades sem problemas.
Em primeira instância foi criado um robô seguidor de linha por meio de circuito básico atuando diretamente na velocidade. Foi de grande importância constatar erros físicos específicos do robô e ambiente, isso por conta dos LDR que são fotoresistêntes, é componente eletrônico passivo do tipo resistor variável, mais especificamente, é um resistor cuja resistência varia conforme a intensidade da luz incide sobre ele.
O interessante desse projeto é a descoberta de problemas físicos que um robô móvel pode sofrer diante a atividades de realização autônoma. Podemos constatar que a variação dos motores em meio a captação de dados do LDR é sujeita a problemas de descontrole parcial do robô.

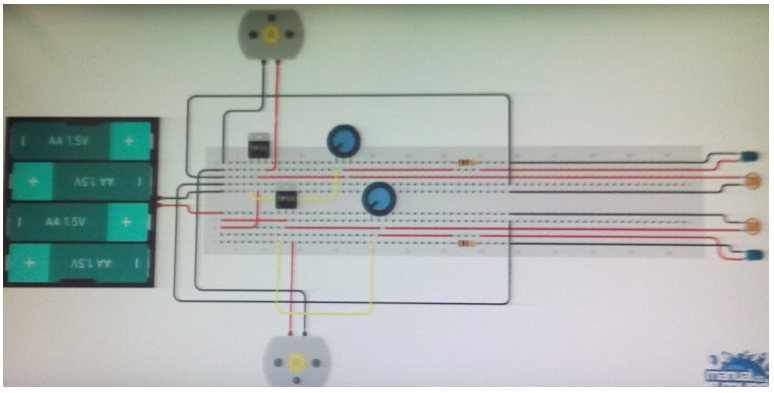


Figura 6: Fonte manual do mundo

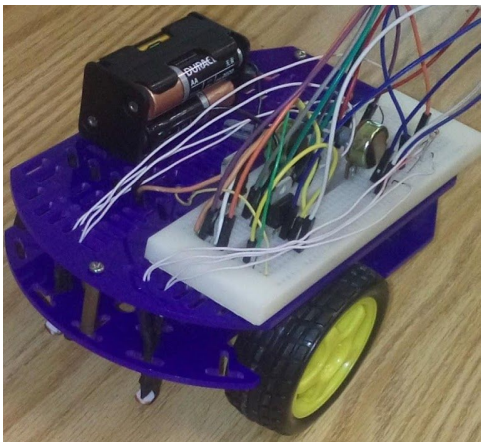


Figura 7:Modelo do robô simples

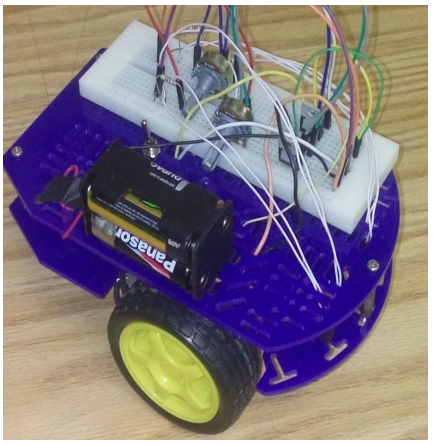


Figura 8:Modelo do robô simples

Robôs seguidores de linha podem ser classificados como um tipo de veículo guiado automaticamente e possuem mecanismos que lhes permitem se movimentar em um ambiente de forma autônoma ou semi autônoma (WOLF et al., 2009).
Com base nos estudos do robô simples foi possível aprimorar um circuito para seguidor de faixa. Esse foi o modelo criado para compor parte do controle.

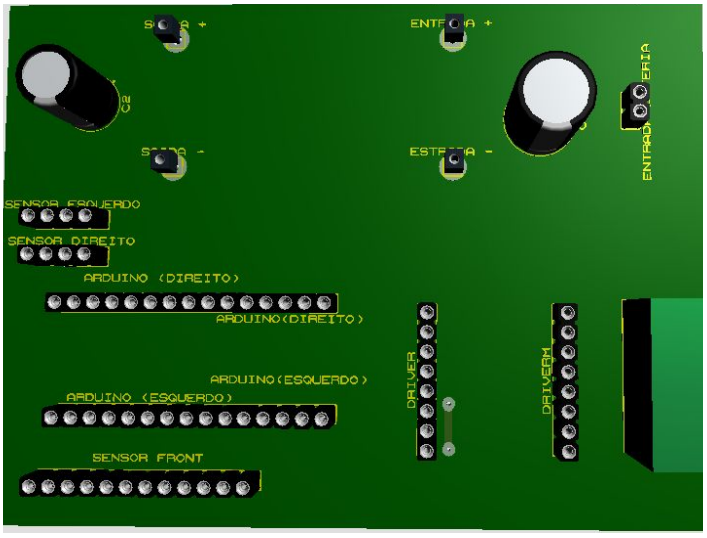


Figura 9:Circuito para seguidor de linha modelo 3D

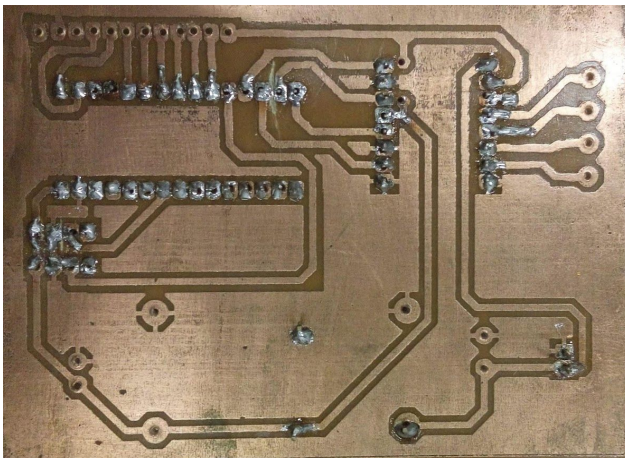


Figura 10:Circuito para seguidor de linha na placa de fenolite

O Proteus Design Suite é um Electronic Design Automation ferramenta (EDA), incluindo captura esquemática, simulação e módulos de layout PCB. O software é executado no do Windows sistema operacional e está disponível em Inglês. Esse software foi de grande ajuda, pois foi desenvolvido a partir dele o modelo da placa de fenolite. No ano de 2017 fiz o curso de instrumentação da plataforma no IFSP-Guarulhos, onde também é fornecido o uso para alunos já que o Proteus é pago.

Uma opção para aqueles que não possuem a licença podem utilizar o KiCad que é um programa computacional de código aberto para projetos de circuitos integrados, com o objetivo de facilitar a concepção de layouts e suas conversões para placas de circuito impresso (PCB). Possui ferramentas para elaboração de estrutura de produtos, arte final e visualizações 3D da PCB e seus componentes.

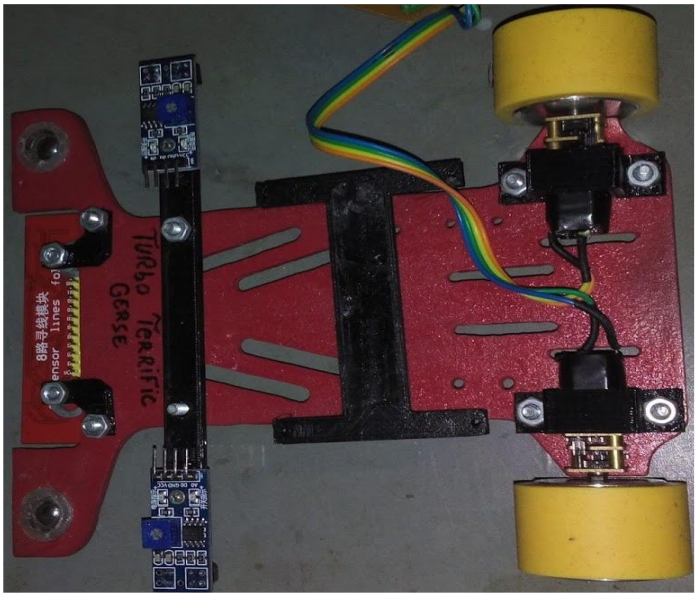


Figura 11:Chassi montado com motores e sensores

O chassi foi utilizado na competição Winter Challenge, com as devidas especificações para uma das maiores competições da área de robótica móvel, esse é sem dúvida um dos maiores eventos de robótica da América Latina. O robô autônomo deve percorrer um circuito feito com uma ou mais placas de MDF revestidas com fórmica preta, tomando como referência uma linha branca de 19±1 mm de largura. O comprimento total da linha é de no máximo 60 metros. Vence o robô que finalizar o trajeto em menor tempo. O corpo do robô deve sempre ficar sobre a linha. Caso o robô saia completamente de cima da linha branca, será considerado que o robô saiu do percurso, assim sendo desclassificado. É a base do robô (parte vermelha), estrutura de suporte para os demais componentes, normalmente feita de acrílico ou qualquer outro material rígido.

Com o uso do software CPN TOOLS foi possível modelar o sistema para seguir linha. O mesmo também é open source, o que facilita o uso dessa poderosa ferramenta. CPN Tools é uma ferramenta para editar, simular e analisar redes de Petri de alto nível. Suporta redes de Petri básicas, além de redes de Petri temporizadas e redes de Petri Coloridas. Tem um simulador e uma ferramenta de análise de espaço de estado incluída.

A CPN Tools foi originalmente desenvolvida pelo Grupo CPN na Universidade de Aarhus de 2000 a 2010. Os principais arquitetos por trás da ferramenta são Kurt Jensen, Søren

Christensen, Lars M. Kristensen e Michael Westergaard. A partir do outono de 2010, a CPN Tools é transferida para o grupo AIS, Eindhoven University of Technology, Holanda.

CPN Tools compreende dois componentes principais, um editor gráfico e um componente simulador backend. O editor gráfico é escrito na linguagem acadêmica, BETA, e o backend do simulador é escrito na variante Standard ML SML / NJ.

A ferramenta apresenta verificação de sintaxe incremental e geração de código, que ocorrem enquanto uma rede está sendo construída. Um simulador rápido que lida eficientemente com redes de tempo indeterminado e temporizados. Espaços de estado completos e parciais podem ser gerados e analisados, e um relatório de espaço de estado padrão contém informações, como propriedades de limitação e propriedades de atividade. Suas funções são de grande utilidade já que demonstra Rede de Petri Colorida com alto desempenho.

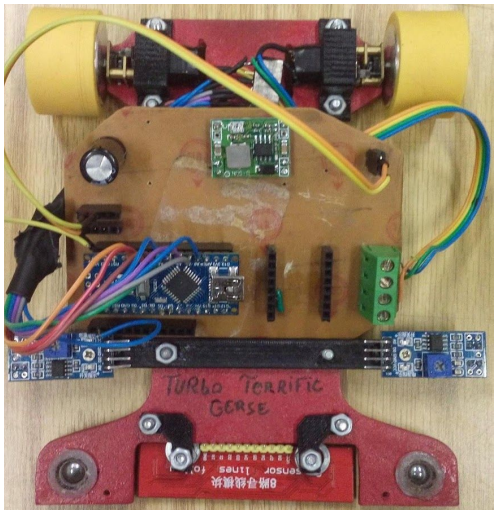


Figura 12: Robô montado com circuito e controlador

RESULTADOS

O problema de robôs móveis tem como um de seus propósitos fazer com que esses sistemas sejam capazes de operar em ambientes com a presença de pessoas (ARMAH; YI; ABU-LEBDEH, 2014).

A ideia por trás da versão de lógica fuzzy é, não apenas distinguir entre preto e branco, mas considerar vários graus de branco, cinza e preto. Nesse aspecto a segurança é muito maior do que condições booleanas. Se o sensor estiver exatamente no limite da faixa, ele verá uma cor entre preto e branco. Quanto mais o sensor fica na linha, mais escura é a cor percebida. Por outro lado, quanto mais o sensor fica na superfície branca, mais branca é a cor percebida.

Agora não estamos mais limitados para virar apenas para a esquerda e para a direita, mas podemos virar mais para a direita (cor mais escura) ou mais para a esquerda (cor mais branca). Desta forma, se exatamente no limite da cor, o robô percorre para frente linearmente.

Para execução do trabalho foi elaborado uma programação no software CPN TOOLS no qual é open source para usuários do mundo inteiro.

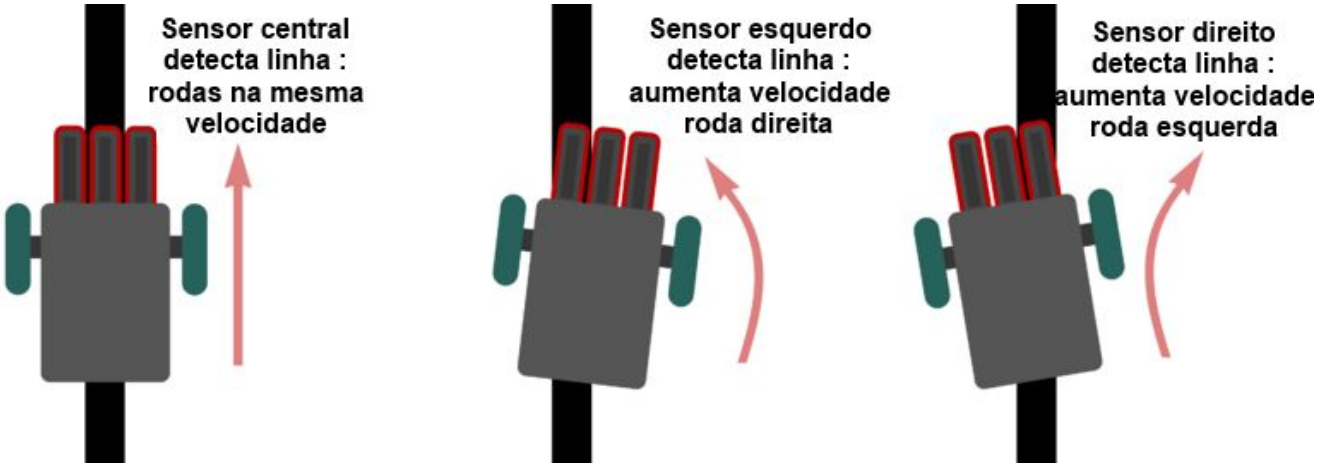


Figura 13: Modelo de ações dos motores(Thomsen,2014)

Utilizou- se dois sensores de refletância:

valor de leitura branco: refletância=80;

valor leitura preto: refletância=20;

Dois motores de valores: 0 parado. 100 veloc. maxima, isso foi acrescentado no software CPN TOOLS junto com o conjunto de regras fuzzy.

4 regras de inferência mostradas a seguir:

R1: Se sensor 1=branco e sensor 2=branco entao motor 1 max vel e motor 2 max veloc

R2: Se sensor 1=preto e sensor 2=branco entao motor 1 para e motor 2 max veloc

R3: Se sensor 1=branco e sensor 2=preto entao motor 1 max vel e motor 2 para

R4: Se sensor 1=preto e sensor2=preto entao motor 1 para e motor 2 para

Sendo assim podemos desenvolver as regras para os demais sensores:

Posição dos sensores na faixa	**Velocidade Motor Esquerdo**	**Velocidade motor Direito**
Sensor4Direita	Parado	Rápido
Sensor3Direita	Devagar	Rápido
Sensor2Direita	Devagar	Rápido
Sensor1Direita	Médio	Rápido
Sensor0Centro	Rápido	Rápido
Sensor1Esquerda	Rápido	Médio
Sensor2Esquerda	Rápido	Devagar
Sensor3Esquerda	Rápido	Devagar
Sensor4Esquerda	Rápido	Parado
Estado fora da faixa	Parado	Parado

Tabela: Regras de inferência fuzzy

Vale ressaltar que regras para estado fora da faixa, na regra está parado por que o robô pode se manter em um loop para encontrar a faixa e ser desclassificado pela competição Winter Challenge . Porém poderíamos colocar uma regra para descrever um gap na pista como na OBR onde o robô segue para frente até encontrar a linha.

Seguindo as bases de inferências podemos constar a situação ativa do robô. O robô não fica em estado on/off devido ao processamento rápido e modelagem para seguir continuamente.

CONCLUSÃO

Conclui-se que o robô inteligente estruturado com a lógica fuzzy é muito mais eficiente em comparação ao tempo de trajeto, sua capacidade para percorrer ambientes desconhecidos é precisa e o mesmo realiza funções como esperado sem sair da pista ou rodear em volta a faixa como alguns robôs seguidores de linha.

REFERÊNCIAS BIBLIOGRÁFICAS

AJ O. Alves. eFLL - Uma Biblioteca fuzzy para arduino e Sistemas Embarcados. Disponível em:http://www.zerokol.com/2012/09/arduinofuzzy-uma-biblioteca-fuzzy-para.html . Acesso em :05/10/2017.

BORBA, Thiago. ARDUINO,LÓGICA FUZZY!!SENSACIONAL!!.Disponível em: http://projetosdoborba.blogspot.com.br/2013/06/arduino-logica-fuzzy-sensacional.html.Acesso em: 15/11/2017.

Dowload ferramenta CPNTOOLS. Disponivel em:<http://cpntools.org/>

Dowload ferramenta KICAD. Disponivel em: <http://kicad-pcb.org/>

ZADEH, Lotfali A. An Introduction to Fuzzy Control,1993.

ZADEH, Lotfali A. Fuzzy sets: information and control, vol. 8, p. 338-353, 1965.

Fuzzy Mathematical Techniques with Applications

J. M. Mendel, Fuzzy Logic Systems for Engineering: a Tutorial, Proc. IEEE, V. 83, No. 3, pp. 345-377, 1995.

ROSS, T. J. Fuzzy Logic with Engenieering Applications. Second Edition.John Wiley & Sons, Ltd. University of New Mexico.

WOLF, Denis. F.; SIMÕES, Eduardo D. V.; OSÓRIO, Fernando S.; TRINDADE JUNIOR, Onofre. Robótica móvel inteligente: da simulação às aplicações no mundo real. In: JAIJornada de Atualização em informática 2009 (Tutorial) – Congresso da SBC, Bento Gonçalves, SP: Editora PUC do Rio, Acesso em: 27 mar. 2018. Disponível em: http://osorio.wait4.org/publications/2009/CL_JAI2009_Completo.pdf

Thomsen,Adilson.Como montar um Robô Seguidor de Linha com Arduino Motor Shield. Disponivel em: https://www.filipeflop.com/blog/projeto-robo-seguidor-de-linha-arduino/800px-vex_line_tracking_graphic-png/. Acesso em 20/12/2017.

Venter, G. e Sobieszczanski-Sobieski, J (2002). Particle Swarm Optimization. Structures, Structural Dynamics and Materials Conference, Denver, Colorado.

Pedersen, M. G. E. (2009) Simplifying Particle Swarm Optimization. University of Southampton, UK. Publicado em Aplied Soft Computing, 2009.

JENSEN, K. et al, "Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems," Int. J. Softw. Tools Technol. Transf., vol. 9, pp. 213-254, 2007.

JENSEN, K. and Kristensen, L., Coloured Petri Nets: Modeling and Validation of Concurrent Systems: Springer-Verlag New York Inc, 2009. MATSUSAKI, C.;

SANTOS FILHO, D. F. Modeling of Distributed Collaborative Control Systems of Production Systems. In: ABCM SYMPOSIUM SERIES IN MECHATRONICS, 2., 2006. [S.L.]. Anais… [S.L.]:ABCM, 2006. p.345-352. MESSIAS, Antônio Rogério.

NAKAMOTO, F. Y. Projeto de sistemas modulares de controle para sistemas produtivos. 2008. 158 p. Tese (Doutorado) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2008. RIZOL,P. et al,. Lógica fuzzy TIPO-2. Revista SODEBRAS, v. 6, p. 27-46, 2011.6. SOMMERVILLE, I. Engenharia de Software. 8th ed. [S.L.]: Addison Wesley, 2007. 568 p.

I. S. Shaw. Fuzzy Control of Industrial Systems: Theory and Applications. Boston, MA: Kluwer Academic Publishers (1998).

N. Ayache (editor). Computer Vision, Virtual Reality, and Robotics in Medicine. Berlin, Germany: Springer-Verlag (1995).

YAMAMOTO, M. M. et al. Um simulador dinâmico para mini-robôs móveis com modelagem de colisões. VI Simpósio Brasileiro de Automação Inteligente., 2003.

MIYAGI, P.E. et al. Petri Net approach for modelling system integration in

intelligent building. Journal of the Brazilian Society of Mechanical Sciences, v. 24, p.

341-230, 2002.

Igei Kaneshiro, P. J .Proposta de um procedimento para a modelagem de sistemas de controle de edifícios inteligentes utilizando a rede de Petri colorida. Doutorado,12/09/2011. Disponível em:http://www.teses.usp.br/teses/disponiveis/3/3141/tde-09122011-095105/en.php. Acesso em: 5/10/2017.

PEDRYCZ, Witold.; GOMIDE, Fernando. Fuzzy systems engineering: toward humancentric computing. IEEE PRESS, 2007.

NOGUEIRA, Maycon M. Aplicando lógica fuzzy no controle de robôs móveis usando dispositivos lógicos programáveis e a linguagem Vhdl. 2013. Dissertação Mestrado em Engenharia Elétrica - Programa de Pós-Graduação em Engenharia Eletrica, UNESP, Ilha Solteira – SP.

JUNG, Claudio Rosito et al. Computação embarcada: Projeto e implementação de veículos autônomos inteligentes. In: XXV CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 2005. Rio Grande do Sul. Anais... Rio Grande do Sul: UNISINOS, 200, p. 2.

ARMAH, S.; YI, S.; ABU-LEBDEH, T. Implementation of autonomous navigation algorithms on two-wheeled ground mobile robot. American Journal of Engineering and Applied Science, p. 16 p., 2014. 16, 20.

APÊNDICE 1: (CAPA CONICT 2018)

APÊNDICE 2: Programação teste dos sensores

testesensores | Arduino 1.8.5

Arquivo Editar Sketch Ferramentas Ajuda

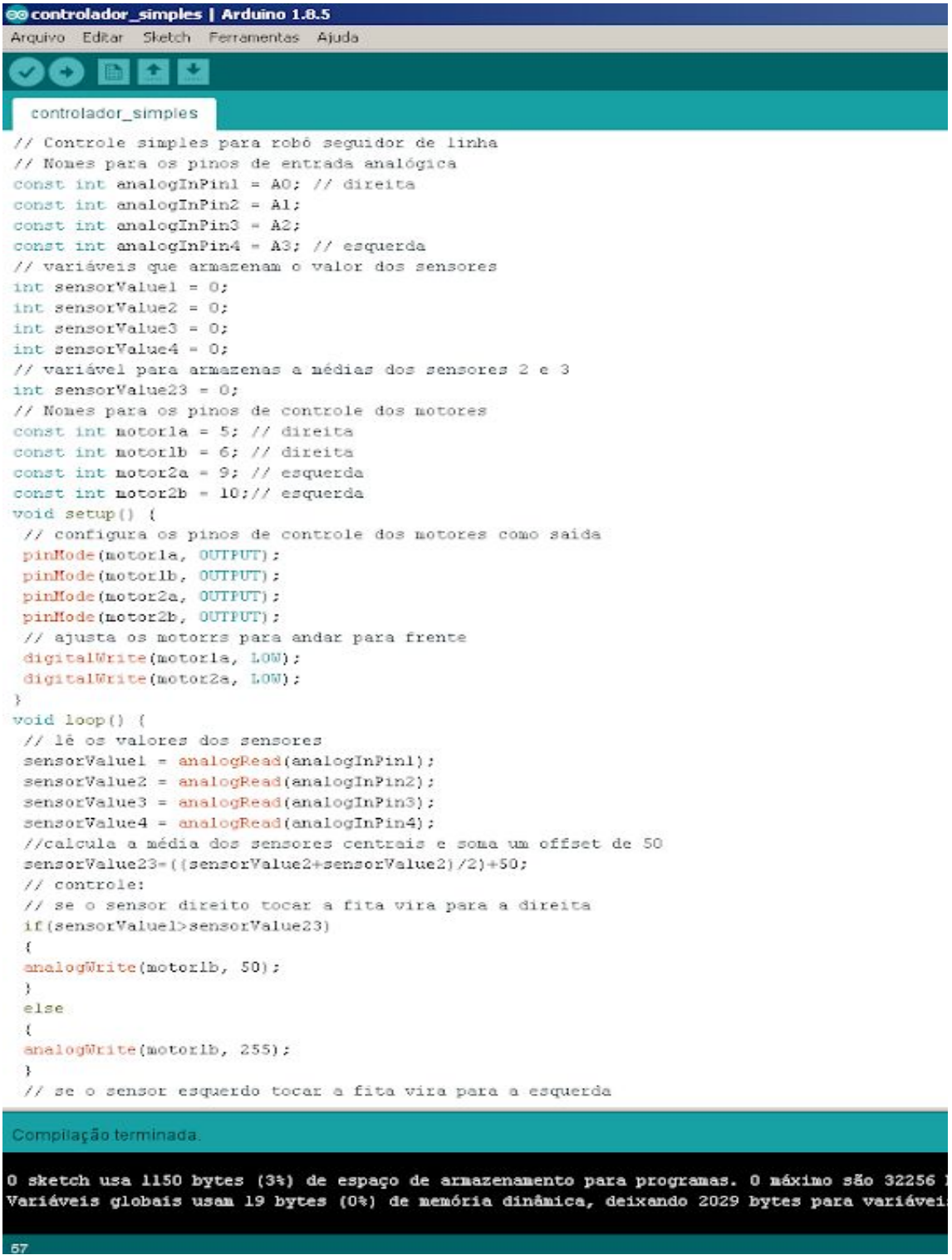
testesensores

```
//Exemplos de códigos de teste do robô
//Teste dos sensores
const int analogInPin1 = A0;
const int analogInPin2 = A1;
const int analogInPin3 = A2;
const int analogInPin4 = A3;
int sensorValue1 = 0;
int sensorValue2 = 0;
int sensorValue3 = 0;
int sensorValue4 = 0;
void setup() {
 // initialize serial communications at 9600 bps:
 Serial.begin(9600);
}
void loop() {
 sensorValue1 = analogRead(analogInPin1);
 sensorValue2 = analogRead(analogInPin2);
 sensorValue3 = analogRead(analogInPin3);
 sensorValue4 = analogRead(analogInPin4);

/*
 Serial.print("sensor1 = ");
 Serial.print(sensorValue1);
 Serial.print("\t ");
 Serial.print("sensor2 = ");
 Serial.print(sensorValue2);
 Serial.print("\t ");
 Serial.print("sensor3 = ");
 Serial.print(sensorValue3);
 Serial.print("\t ");
 Serial.print("sensor4 = ");
 Serial.print(sensorValue4);
 Serial.print("\n");
 */
 Serial.print(sensorValue1);
 Serial.print("\t ");
 Serial.print(sensorValue2);
 Serial.print("\t ");
 Serial.print(sensorValue3);
 Serial.print("\t ");
 Serial.print(sensorValue4);
 Serial.print("\n");

 delay(2);
}
```

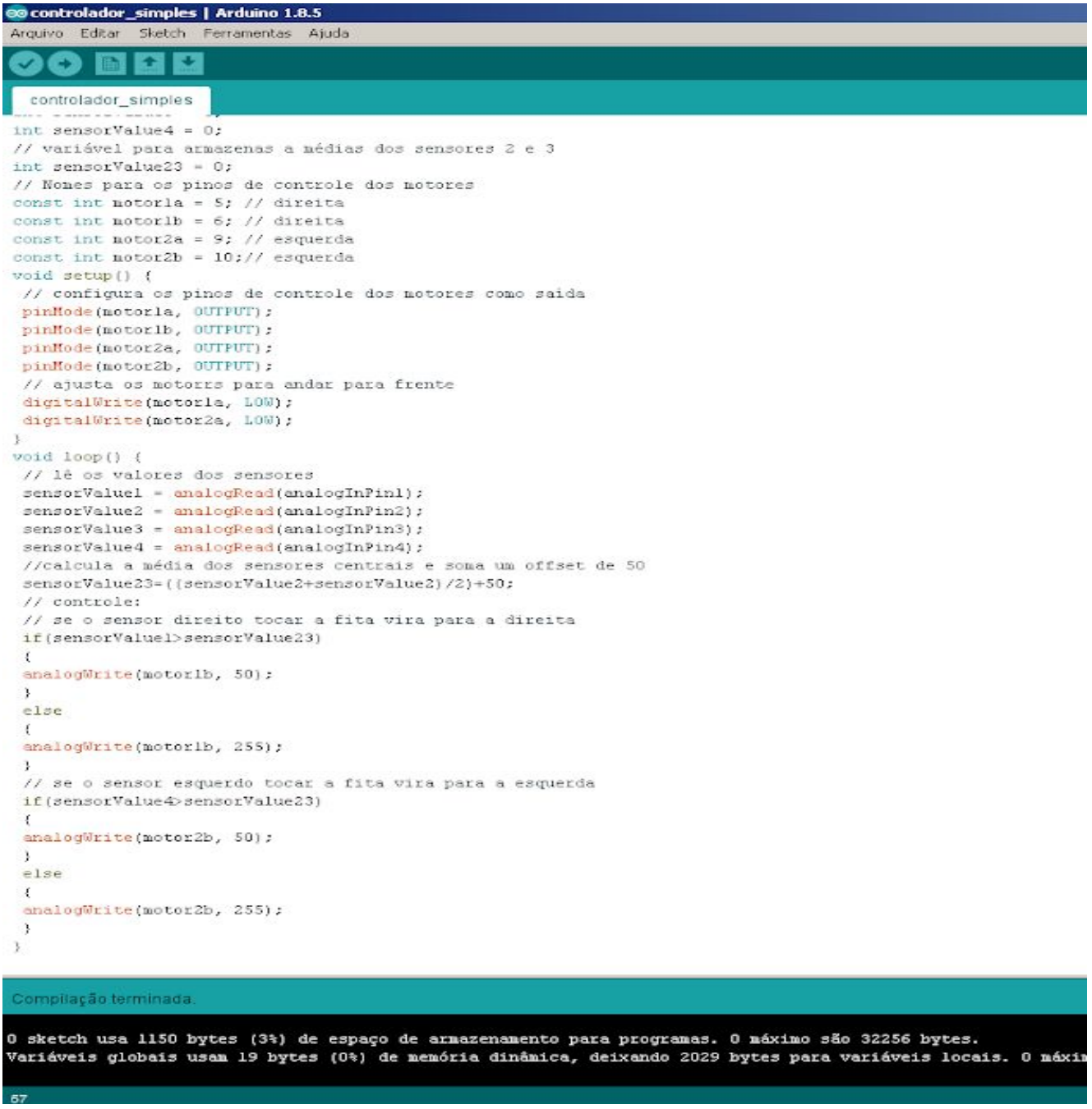
Compilação terminada.

```
O sketch usa 2114 bytes (6%) de espaço de armazenamento par
Variáveis globais usam 198 bytes (9%) de memória dinâmica,
```

APÊNDICE 3: Programação para robô seguidor de linha

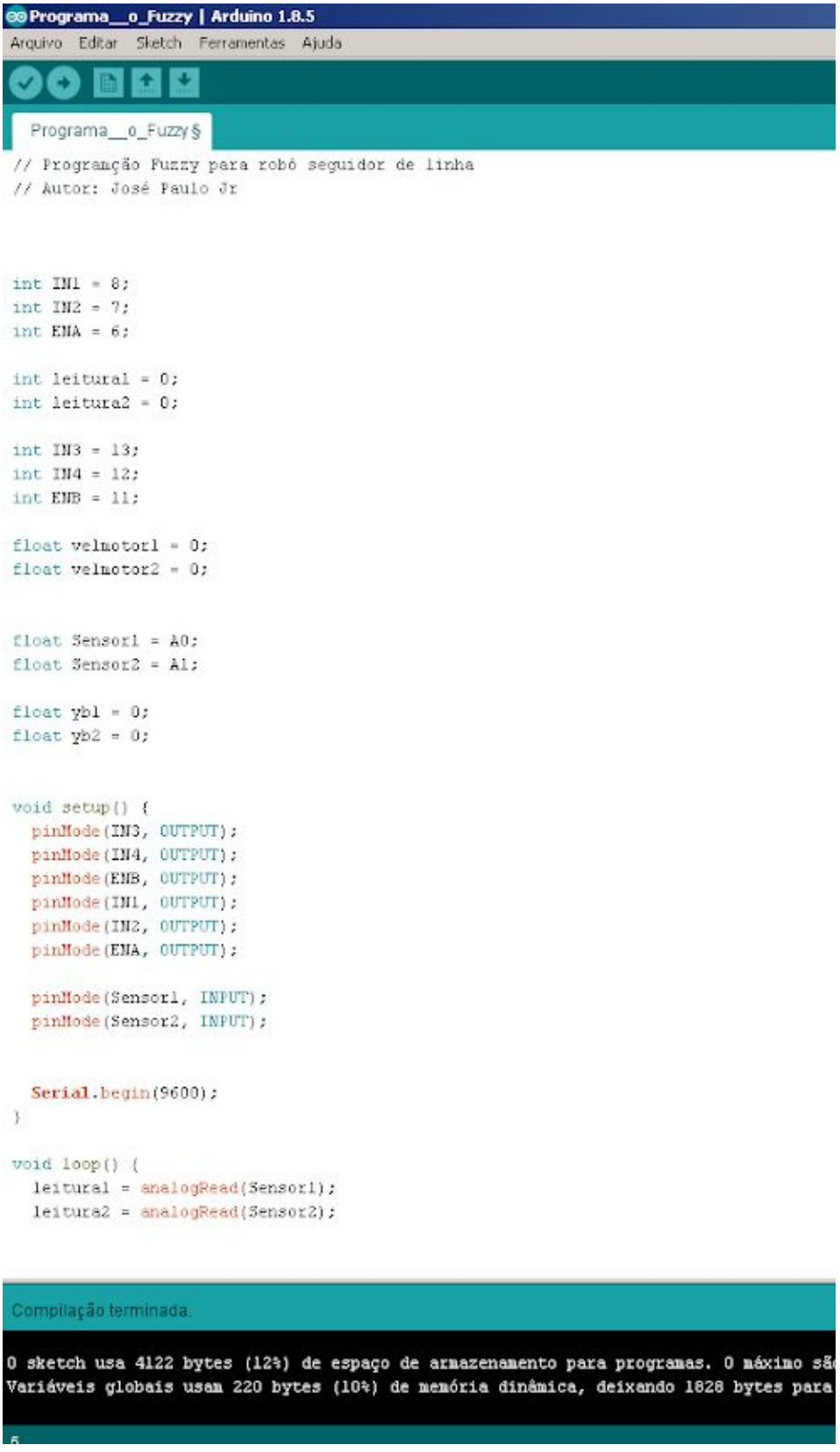


continuação…


controlador_simples | Arduino 1.8.5
Arquivo Editar Sketch Ferramentas Ajuda
controlador_simples
int sensorValue4 = 0;
// variável para armazenas a médias dos sensores 2 e 3
int sensorValue23 = 0;
// Nomes para os pinos de controle dos motores
const int motor1a = 5; // direita
const int motor1b = 6; // direita
const int motor2a = 9; // esquerda
const int motor2b = 10;// esquerda
void setup() {
// configura os pinos de controle dos motores como saida
pinMode(motor1a, OUTPUT);
pinMode(motor1b, OUTPUT);
pinMode(motor2a, OUTPUT);
pinMode(motor2b, OUTPUT);
// ajusta os motorrs para andar para frente
digitalWrite(motor1a, LOW);
digitalWrite(motor2a, LOW);
}
void loop() {
// lê os valores dos sensores
sensorValue1 = analogRead(analogInPin1);
sensorValue2 = analogRead(analogInPin2);
sensorValue3 = analogRead(analogInPin3);
sensorValue4 = analogRead(analogInPin4);
//calcula a média dos sensores centrais e soma um offset de 50
sensorValue23=((sensorValue2+sensorValue2)/2)+50;
// controle:
// se o sensor direito tocar a fita vira para a direita
if(sensorValue1>sensorValue23)
{
analogWrite(motor1b, 50);
}
else
{
analogWrite(motor1b, 255);
}
// se o sensor esquerdo tocar a fita vira para a esquerda
if(sensorValue4>sensorValue23)
{
analogWrite(motor2b, 50);
}
else
{
analogWrite(motor2b, 255);
}
}
Compilação terminada.
0 sketch usa 1150 bytes (3%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 19 bytes (0%) de memória dinâmica, deixando 2029 bytes para variáveis locais. O máxi
67

fim.

APÊNDICE 4: Programação fuzzy NO ARDUINO

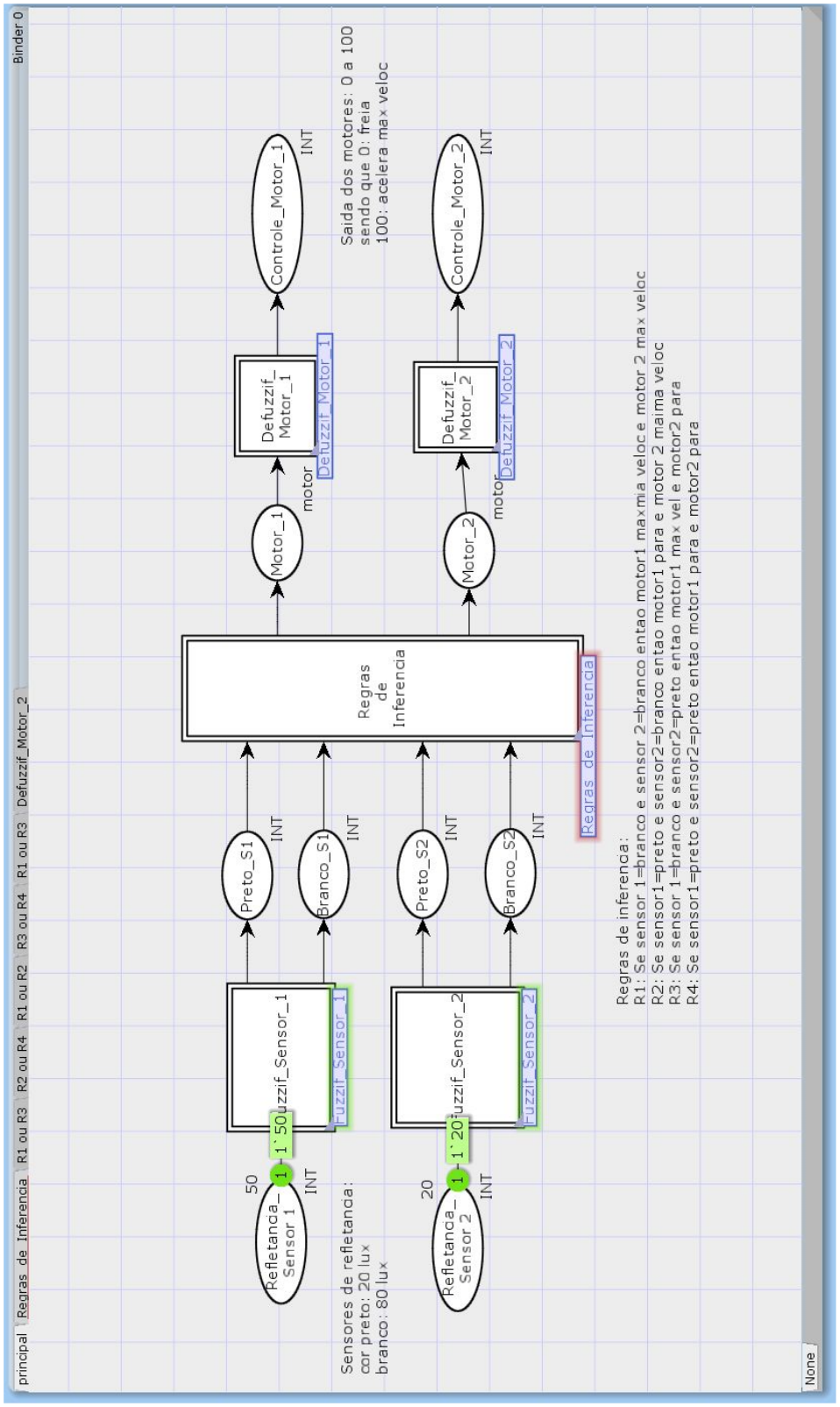


continuação…


Programa__o_Fuzzy | Arduino 1.8.5
Arquivo Editar Sketch Ferramentas Ajuda
Programa__o_Fuzzy §
void setup() {
pinMode(IN3, OUTPUT);
pinMode(IN4, OUTPUT);
pinMode(ENB, OUTPUT);
pinMode(IN1, OUTPUT);
pinMode(IN2, OUTPUT);
pinMode(ENA, OUTPUT);
pinMode(Sensor1, INPUT);
pinMode(Sensor2, INPUT);
Serial.begin(9600);
}
void loop() {
leitura1 = analogRead(Sensor1);
leitura2 = analogRead(Sensor2);
yb1 = (827 - leitura1)* 0.1176470588;
yb2 = (827 - leitura2)* 0.1176470588;
velmotor1 = yb1 * 1.98;
velmotor2 = yb2 * 1.98;
Serial.println(leitura1);
Serial.println(leitura2);
Serial.println(velmotor1);
Serial.println(velmotor2);
digitalWrite(IN3, LOW);
digitalWrite(IN4, HIGH);
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
analogWrite(ENB, velmotor2);
analogWrite(ENA, velmotor1);
}
Compilação terminada.
O sketch usa 4122 bytes (12%) de espaço de armazenamento para programas. O máximo são 32256 bytes.
Variáveis globais usam 220 bytes (10%) de memória dinâmica, deixando 1828 bytes para variáveis locais. O máximo são

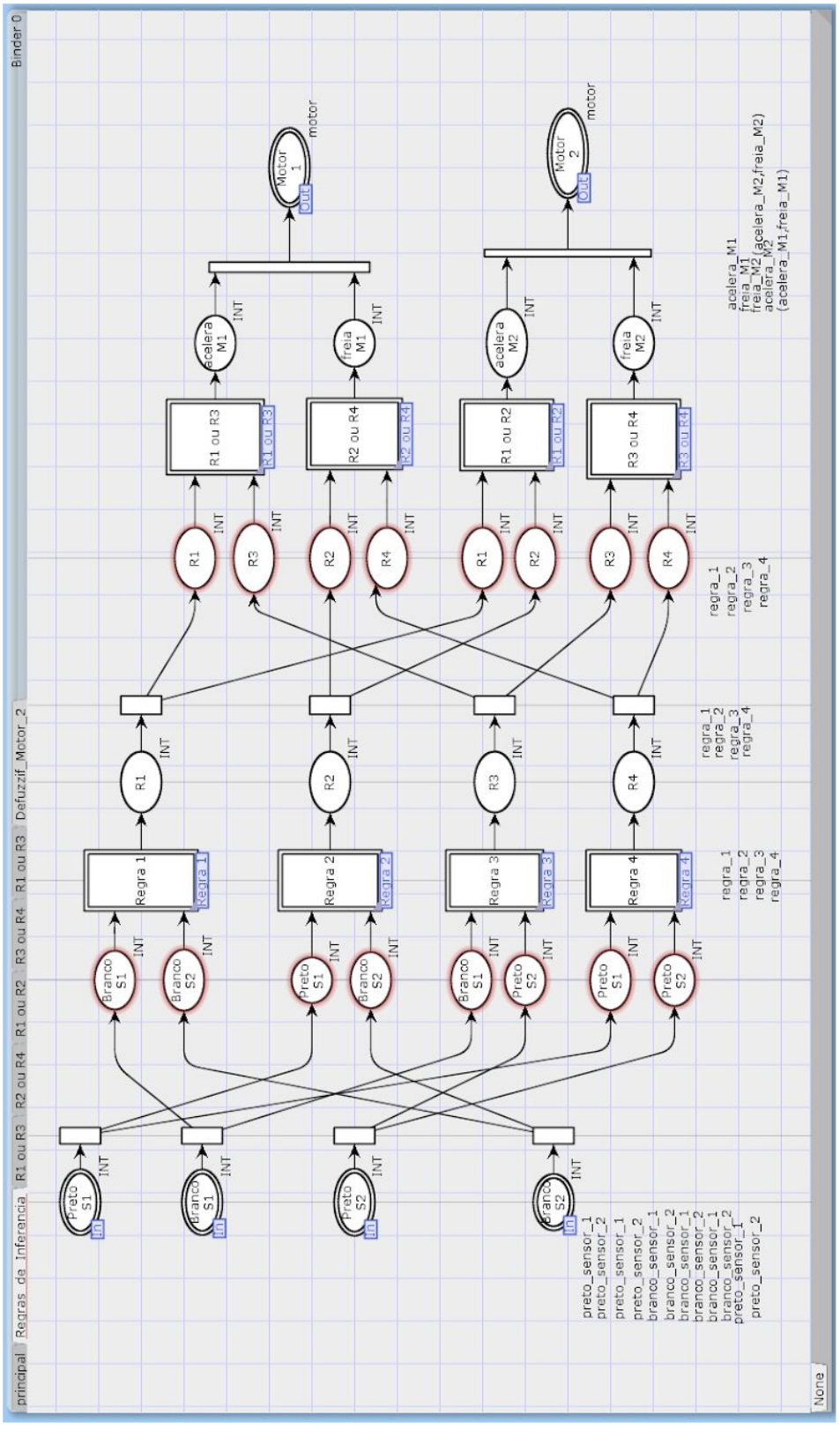
Fim.

APÊNDICE 5: Programação no CPNTools que utiliza lógica fuzzy para controlar um carrinho seguidor de linha



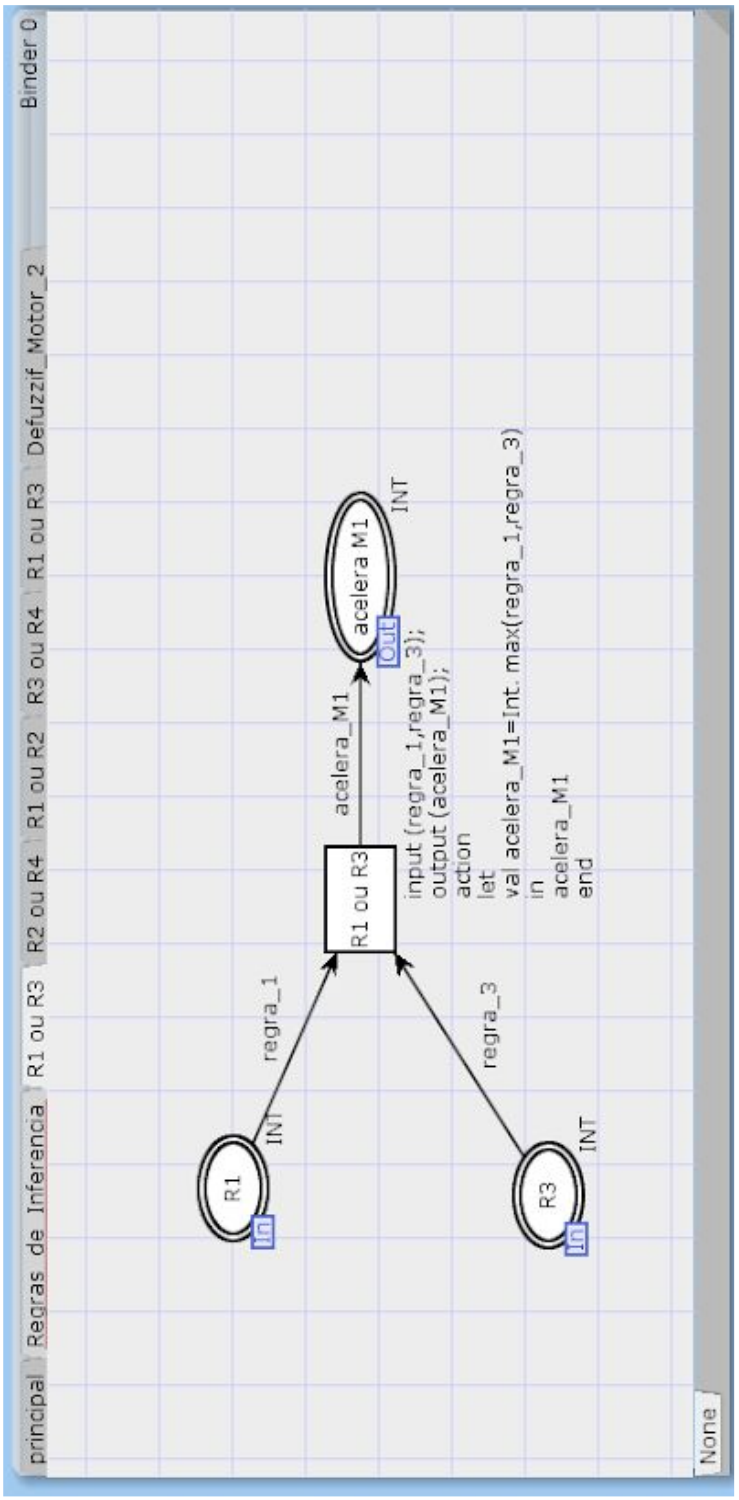
Nesse modelo está organizada a modelagem fuzzy principal para carrinho seguidor de linha onde o processo fuzzy está desenvolvido. Presente estão os valores crispi de entrada para inferência as condições físicas dos sensores seguidas das regras de inferência, os atuadores que são os motores trabalham de acordo com o processo anterior por serem controlados de acordo com as saídas. Caso o sensor 1 ler 50 lux e o sensor 2 ler 70 lux, esses valores passarão pelas condições e regras de inferência que desenvolvem saídas para a potência dos motores no qual desempenharam o trajeto da faixa. Neste caso o motor esquerdo irá parar e o motor direito andar para fazer uma curva à direita.

Continua...



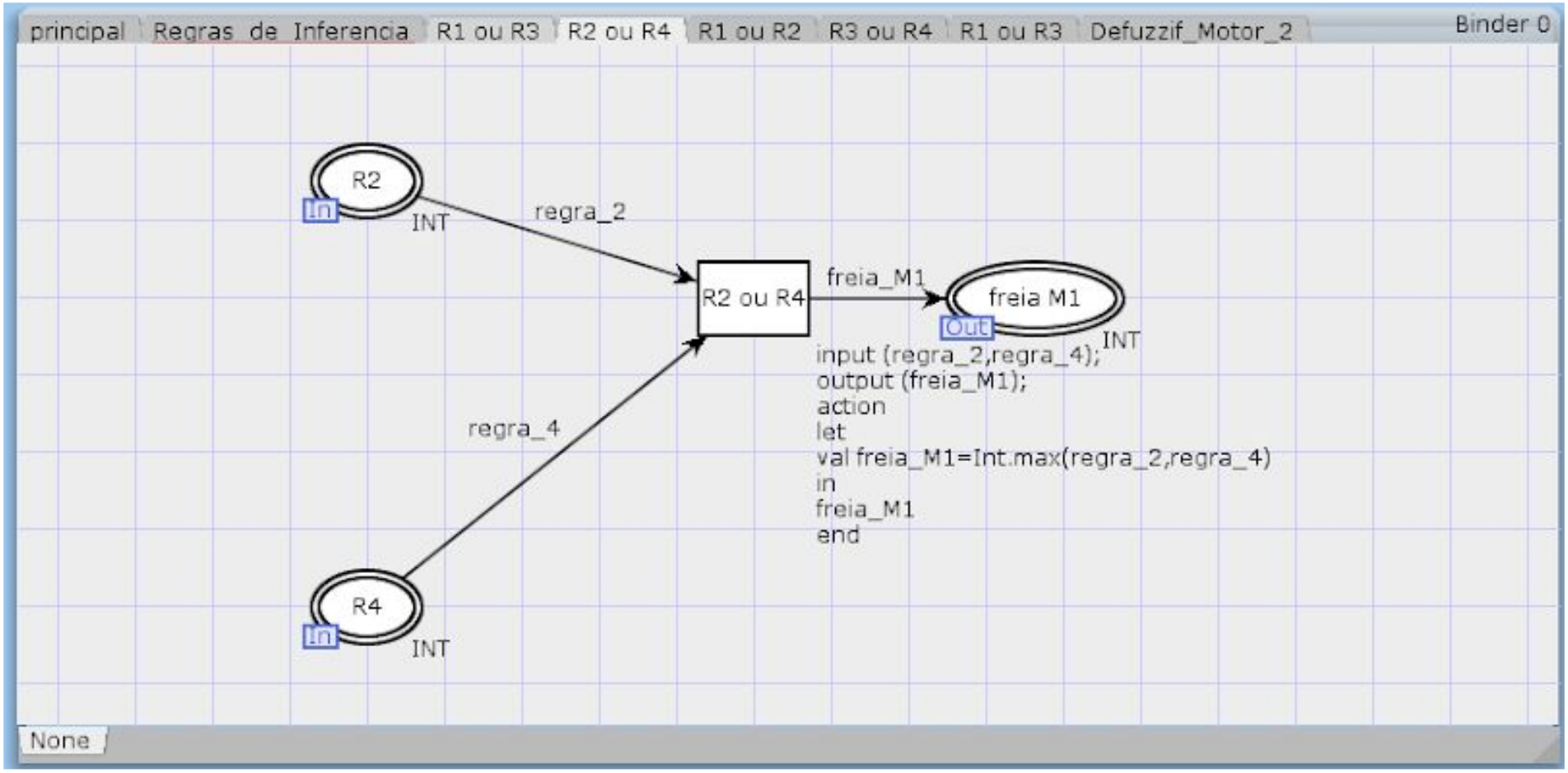
De acordo com as regras de inferências temos o processo de modelagem fuzzy.

Processo de tomada de decisão mediante a conjuntos fuzzy.



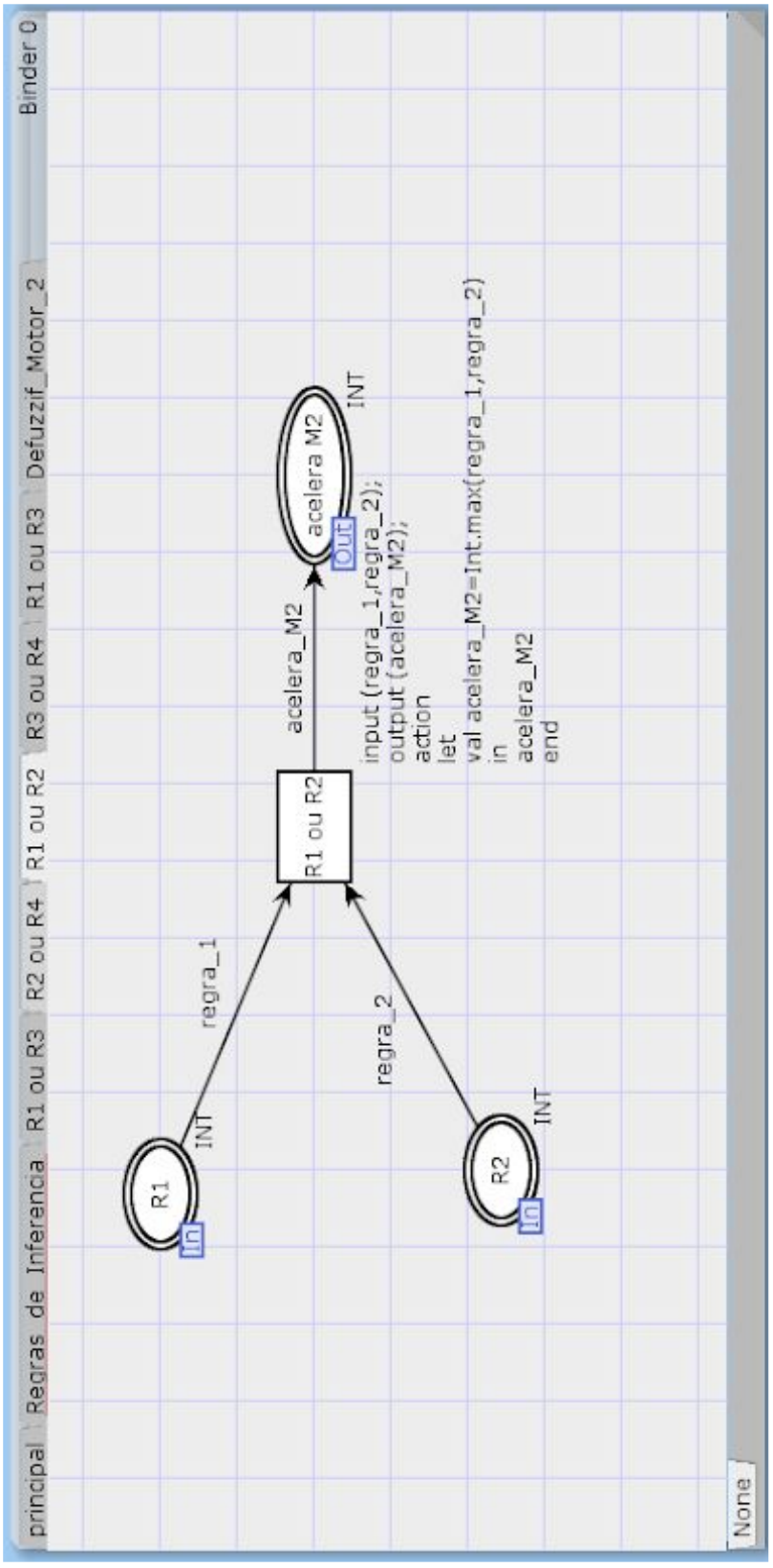
Entrada de regras para inferência.

Continua...


principal
Regras_de_Inferencia
R1 ou R3
R2 ou R4
R1 ou R2
R3 ou R4
R1 ou R3
Defuzzif_Motor_2
Binder 0
R2
In
INT
regra_2
R2 ou R4
freia_M1
freia M1
Out
INT
input (regra_2,regra_4);
output (freia_M1);
action
let
val freia_M1=Int.max(regra_2,regra_4)
in
freia_M1
end
regra_4
R4
In
INT
None

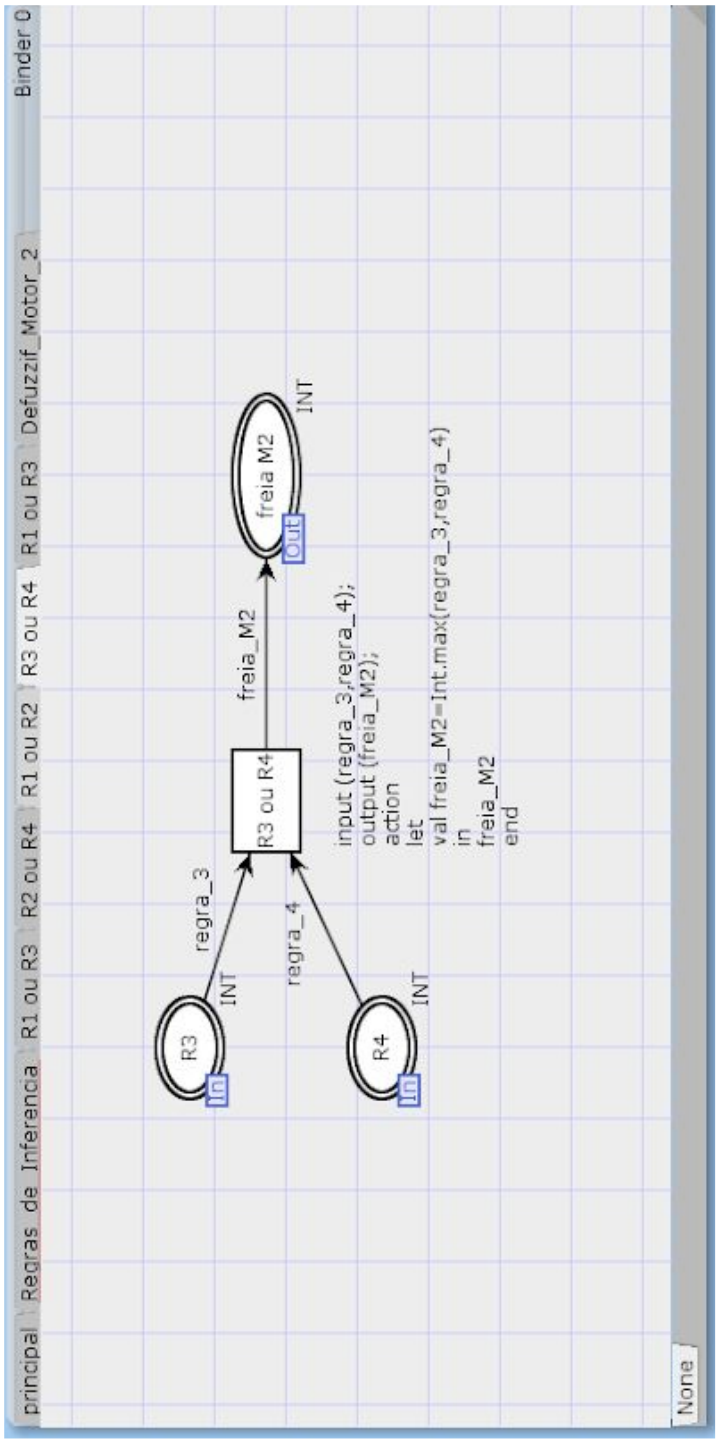
Entrada de regras para inferência.

Continua...



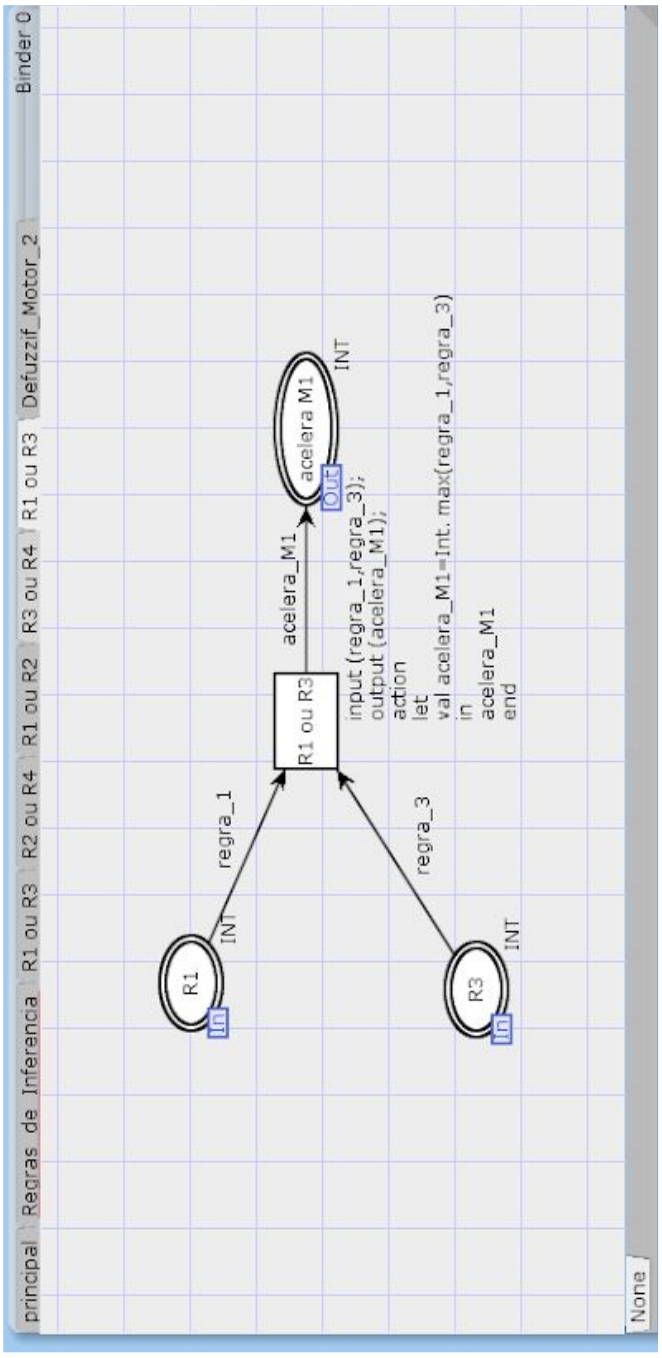
Entrada de regras para inferência.

Continua...



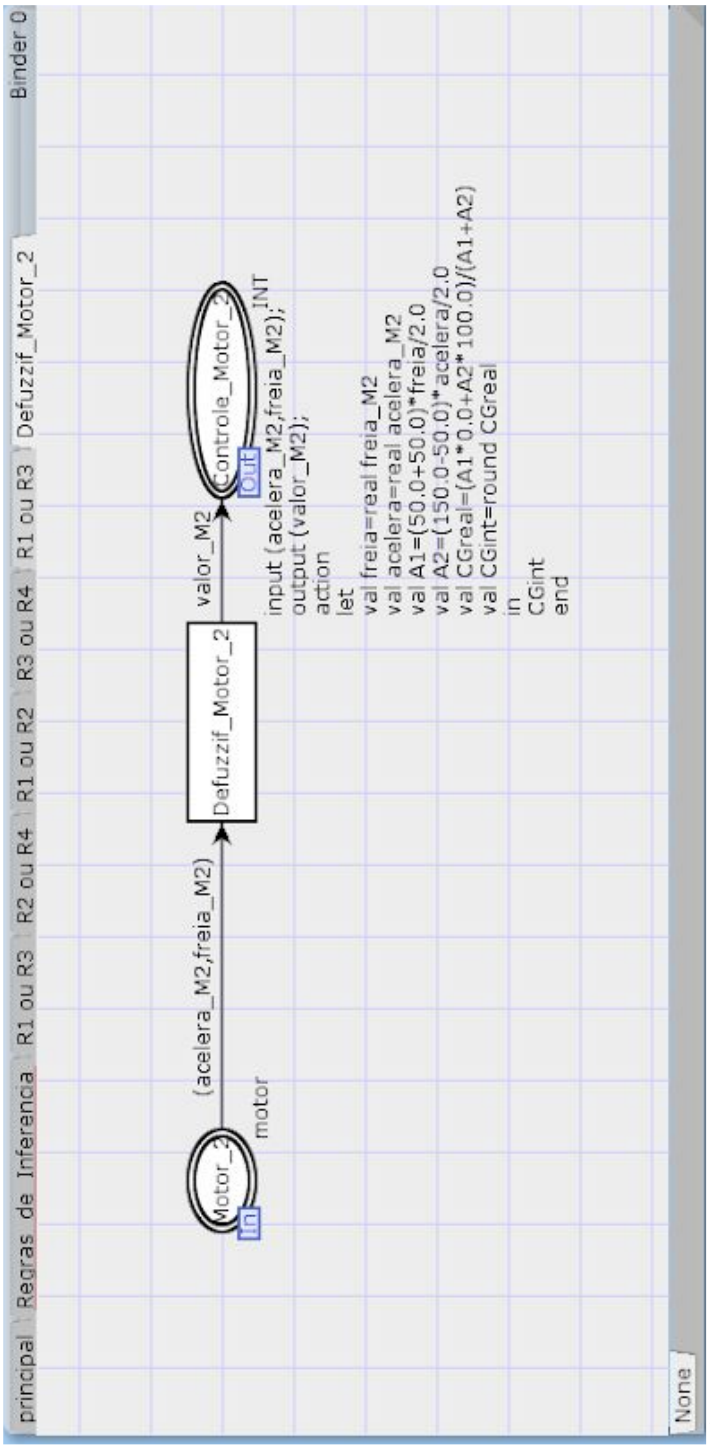
Entrada de regras para inferência.

Continua...

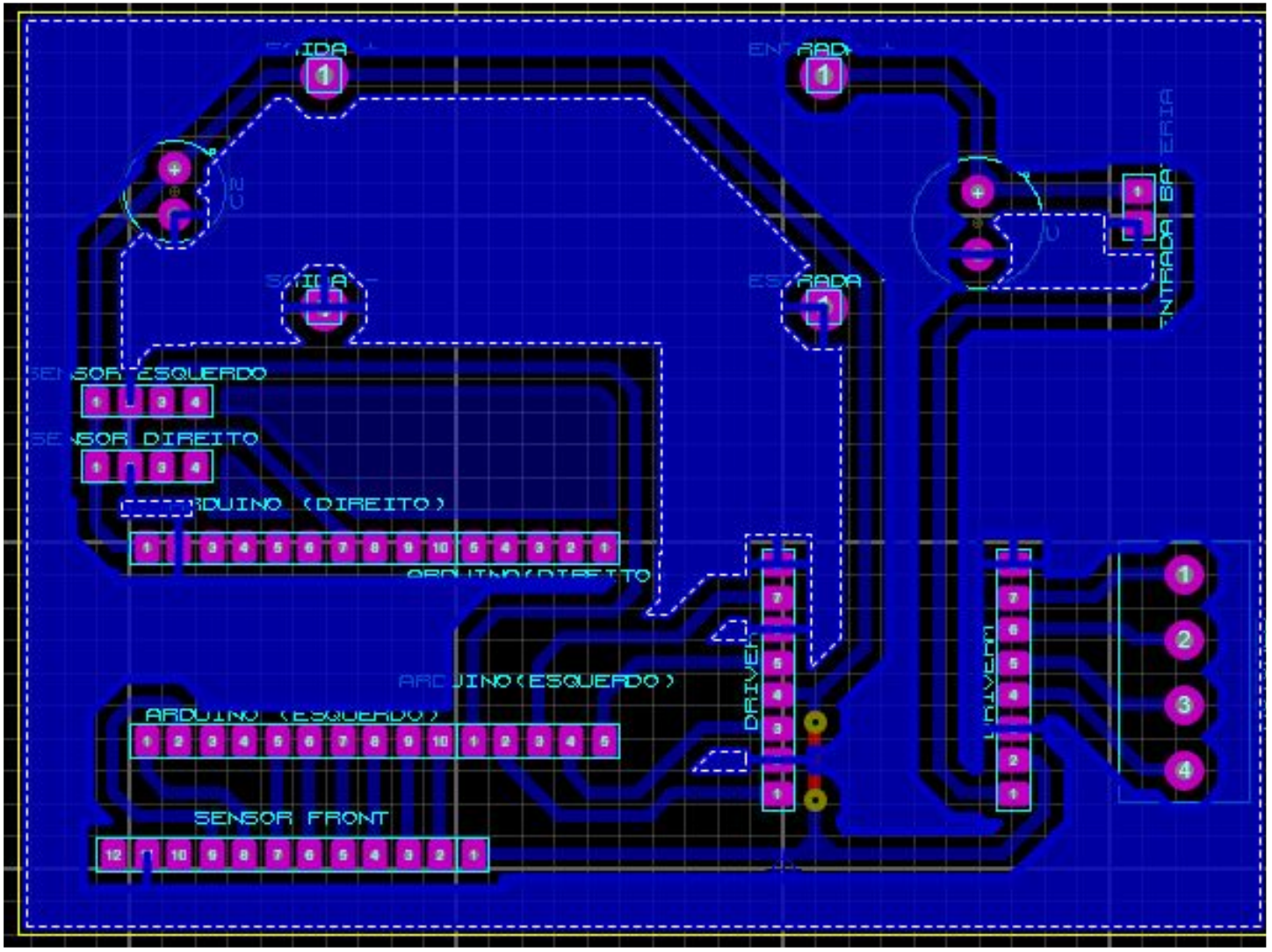


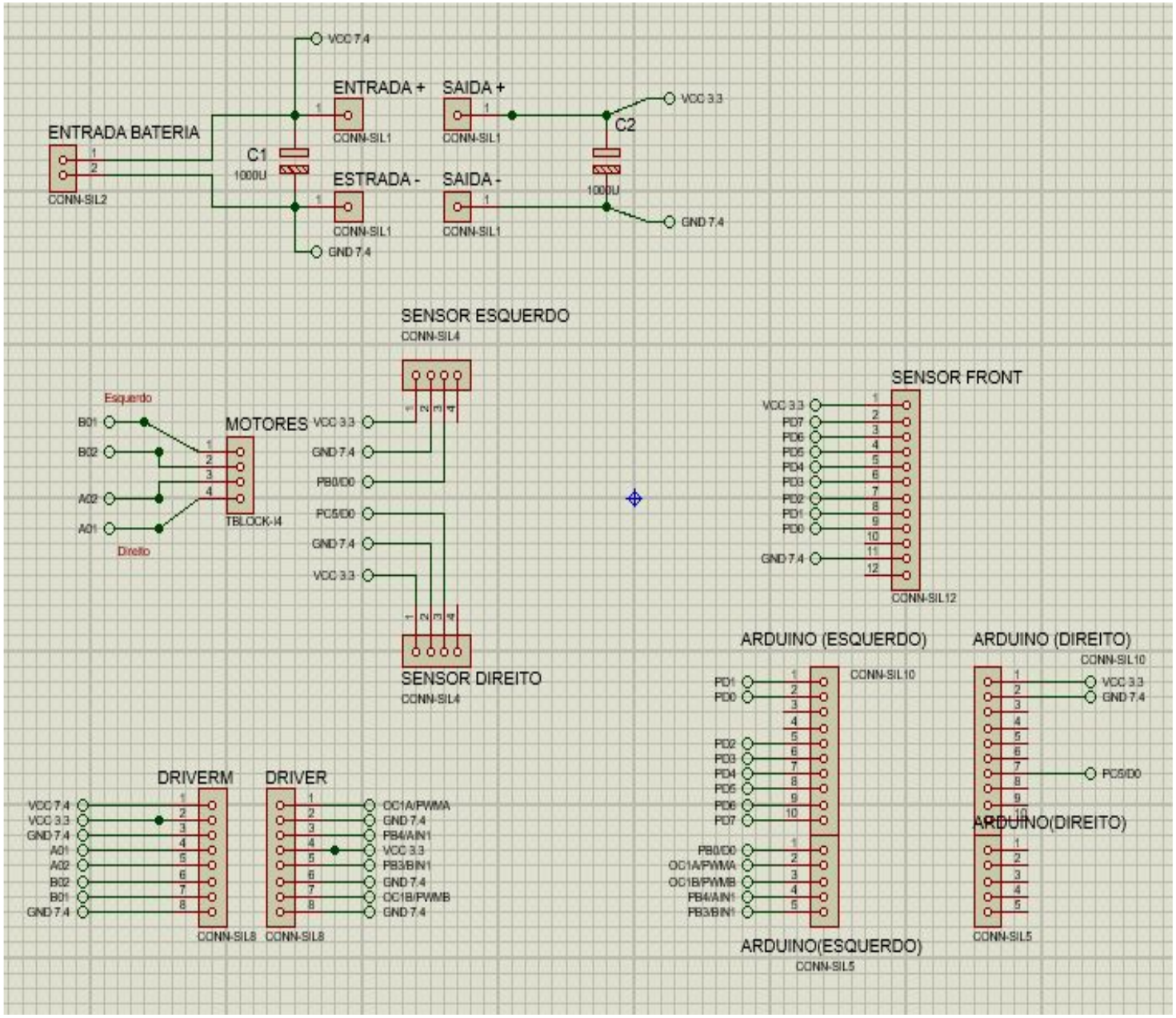
Entrada de regras para inferência.

Continua…



Valores de saídas pela defuzzificação, com potência dos motores definidas.Fim


SENSOR ESQUERDO
SENSOR DIREITO
ARDUINO (DIREITO)
ARDUINO(ESQUERDO)
ARDUINO (ESQUERDO)
SENSOR FRONT
DRIVER
DRIVER
ENTRADA BATERIA


VCC 7.4
ENTRADA +
SAIDA +
VCC 3.3
ENTRADA BATERIA
C1
1000U
C2
1000U
ESTRADA -
SAIDA -
GND 7.4
CONN-SIL1
CONN-SIL2
SENSOR ESQUERDO
CONN-SIL4
SENSOR FRONT
CONN-SIL12
Esquerdo
Direito
B01
B02
A02
A01
MOTORES
TBLOCK-I4
PB0/D0
PC5/D0
SENSOR DIREITO
ARDUINO (ESQUERDO)
ARDUINO (DIREITO)
CONN-SIL10
ARDUINO(DIREITO)
ARDUINO(ESQUERDO)
CONN-SIL5
DRIVERM
DRIVER
CONN-SIL8
OC1A/PWM/A
OC1B/PWM/B
PB4/AIN1
PB3/BIN1