DESENVOLVIMENTO DE UMA ARQUITETURA PARA CONTROLE DIRECIONAL DE VOO DE UM VEÍCULO AÉREO NÃO-TRIPULADO BASEADO EM UMA TÉCNICA DE INTELIGÊNCIA ARTIFICIAL

Tamirys V. Ribeiro[1]
Marco F. de Sousa[2]

RESUMO

Este trabalho tem como objetivo realizar as aplicações dos principais simuladores destinados ao controle de veículos aéreos não-tripulados para aplicações que têm se tornado relevantes na área de segurança, indústria, agricultura, meio ambiente, fotografia aérea e filmagem. Objetivou-se compreender as aplicações citadas em referências especializadas a fim de obter o desenvolvimento de uma arquitetura virtual de um *quadcopter* utilizando lógica *fuzzy*, em um simulador V-REP. Neste trabalho, serão estudados os veículos aéreos não tripulados, e controladores *fuzzy*, além do simulador V-REP e a arquitetura de desenvolvimento da aplicação para um controle autônomo do VANT.
Palavras-chaves: *Quadcopter*. Inteligência Artificial. Arquitetura. Voo Autônomo.

ABSTRACT

This work aims to realize the applications of the main simulators destined to control unmanned aerial vehicles for applications that have become relevant in the area of safety, industry, agriculture, environment, aerial photography and filming. In addition to results obtained through a research that aimed to understand the applications cited in specialized references in order to obtain a development of a virtual architecture of a quadcopter using fuzzy logic, in a V-REP simulator. In this work will be contextualized on unmanned aerial vehicles, and fuzzy controllers. In addition to a general context about the V-REP simulator and application development architecture for a stand-alone UAV control.
Keywords: Quadcopter, Artificial Intelligence, Architecture, Autonomous Flying.

1. INTRODUÇÃO

Dentre diversas particularidades, a autonomia do veículo aéreo não-tripulado - *VANT* tem ganhado destaque e foco de pesquisa de sistemas de controle embarcados. O uso de VANT se torna relevante nas áreas relacionadas à agricultura, bem como em outros domínios de aplicações, como aponta Jenkins (2014).

[1] Acadêmica do curso de Sistema de Informação da Faculdade Católica do Tocantins. E-mail virgulinotamirys@gmail.com
[2] Professor do curso de Sistema de Informação da Faculdade Católica do Tocantins. E-mail marco@catolica-to.edu.br

As aplicações nas quais esta pesquisa é inserida buscam concentrar seu foco em tarefas de missões de localização e tática. Nesse sentido, a comunicação e a possibilidade de colaboração em diferentes ambientes devem ser abordadas. Apesar do universo de dispositivos aéreos não tripulados, esta pesquisa limitará sua aplicação aos chamados *quadcopters*, um veículo dotado de 4 motores em uma estrutura frame em X, tendo um motor em cada ponta do X. Este veículo possui a capacidade de decolar e aterrissar na vertical, sendo um VTOL (*Vertical Take and Landing*).

O objetivo deste trabalho é criar uma aplicação simulada que integrará componentes e técnicas de melhorias de raciocínio e simuladores com *API,* integrando-os. Assim, constatando que os componentes estão armazenando conhecimentos.

Deste modo, a simulação trará dados visuais que poderão ser analisados, sendo possível definir se a arquitetura utilizada para aplicação dos componentes é viável ou não.

2. METODOLOGIA

Esta pesquisa trata-se de um estudo de caso, no qual aplicou-se o método descritivo, numa abordagem qualitativa no desenvolvimento da arquitetura para tratamentos de trajetórias, utilizando inteligência artificial, enfatizando uma combinação de todas as relações provenientes das diversas regras e com características de aprendizagem e adaptação de um VANT real/simulado.

Quanto à natureza da pesquisa, trata-se de pesquisa aplicada, pois buscou-se entender sobre o funcionamento do controle de um VANT, com objetivos de aplicar possíveis ações, podendo ter controle de tarefas do veículo com aplicações simuláveis e no mundo real. Quanto aos procedimentos, teve orientação no estudo de caso do projeto sustentação na revisão da literatura específica, como: análise de livros, perfil do VANT e aplicações de arquitetura.

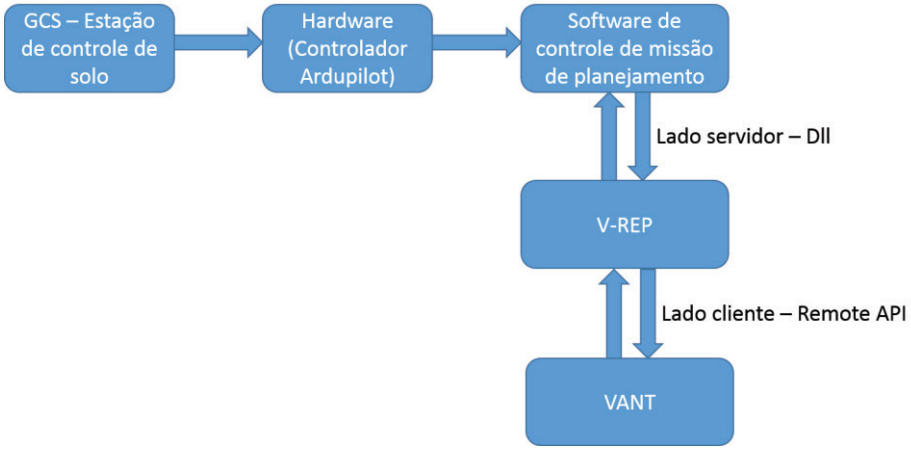
Os procedimentos metodológicos adotados tendem a responder a motivação principal do simulador: validar a capacidade de controle de missão do VANT e de atribuir missões a ele. Toda estrutura da simulação pode ser representada, constando seu funcionamento em *libs* utilizadas e trocas de informações do simulador e da plataforma.

Após o estudo, foi efetuada uma comparação levantando características de

possíveis arquiteturas sem modelo de raciocínio, sem uma integração entre os principais interlocutores da arquitetura, representadas nas Figuras 1 e 2.

A arquitetura é representada na figura 1, demonstrando a relação de GCS: estações de controle de solo com o controlador do *Ardupilot* (*Hardware*), interligado com o software que passará a missão de planejamento do simulador para o *VANT* via *API Remote*.

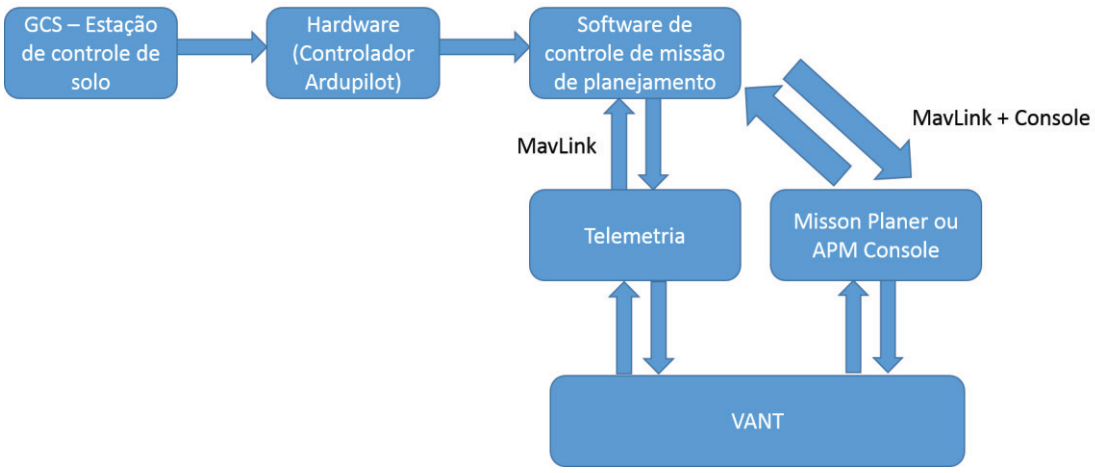
Figura 1 - Arquitetura API REMOTE



Fonte: Produzido pelos autores

A figura 2, representa a relação de GCS: estações de controle de solo com o controlador do *Ardupilot* (*Hardware*), interligado com o software que passara a missão de planejamento via telemetria e *misson planer* para o *VANT*.

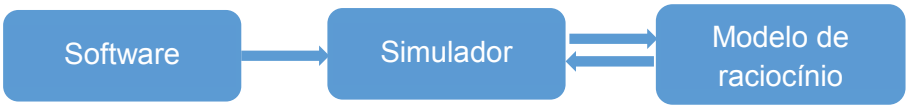
Figura 2 - Arquitetura Telemetria



Fonte: Produzido pelos autores

De acordo com essa comparação, foi analisada a necessidade de existir um modelo de raciocínio. Uma arquitetura que possua um software de controle, o simulador e o modelo de raciocínio que será a aplicação da lógica fuzzy para representar o comportamento próximo do humano.

Figura 3 - Arquitetura Lógica Fuzzy



Fonte: Produzido pelos autores

Portanto, como software de controle, terá o *Ardupilot*, além do simulador V-REP e como modelo de raciocínio a lógica fuzzy. A real arquitetura será ilustrada posteriormente, com mais detalhes.

Foram realizadas várias arquiteturas no decorrer do desenvolvimento da pesquisa. Porém, de alguma forma, nenhuma atendia por completo as necessidades para ser a arquitetura utilizável no processo de desenvolvimento deste estudo de caso.

Atualmente, de acordo com pesquisas, tem-se o *firmware*, que é o *ardupilot*, muito bom no que se refere ao aspecto de controle e a parte de aplicação para execução do mesmo. E o V-REP muito bom para simulação quando se trata de não ter o físico do hardware. O faltante é o integrador de ambos, um módulo, ou algo que já vem nativo. Nessa nova arquitetura, essa integração será solucionada.

3. VEÍCULOS AÉREOS NÃO-TRIPULADO (*VANT*)

Revisou-se os conceitos sobre *VANTs*, baseado na dissertação sobre Controle de Missão de Voo de Veículo Aéreo Não-Tripulado (BEHNCK, 2014).

Os *VANT's* são aeronaves que podem voar com ausência de tripulação, podendo ser controlados remotamente. Atendem a fins comerciais e profissionais, com objetivo de pesquisa e experimentos. Há uma grande diversidade de modelos *VANT's*, com maioria focada na parte civil.

A utilização de *VANTs* traz algumas vantagens em relação às operações militares que desempenham missões de riscos sem colocar a tripulação em perigo.

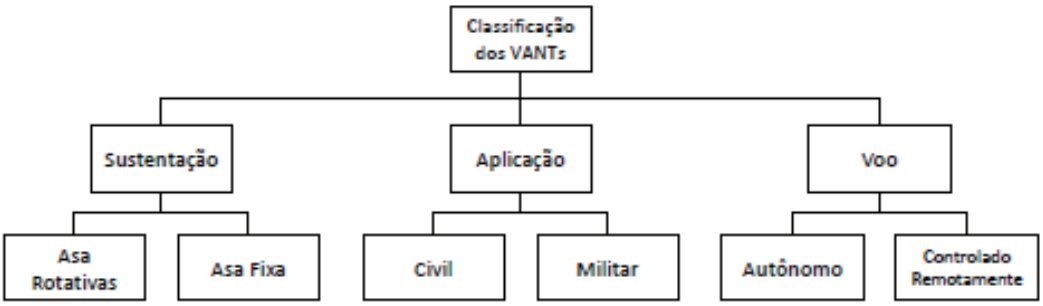
Podem, ainda, desempenhar atividades de vigilância, em locais nos quais haja contaminação química ou radioativa.

Existem características do módulo estudado, seus fundamentos, que estão presentes em quase todos os modelos, como o sistema de navegação, utilizando navegação autônoma, ou remotamente. Além disso, as aeronaves podem utilizar meios de comunicação para enviar fotografias e dados coletados da missão.

Existem outras características que são totalmente dependentes da aplicação, como as dimensões da aeronave, a capacidade de carga e componentes acoplados.

Os conceitos sobre *VANTs* podem ser ilustrados resumidamente por meio da Figura 4: Classificação dos Vants.

Figura 4: Classificação dos Vants. Unmanned Aircraft Systems (AUSTIN, 2011).



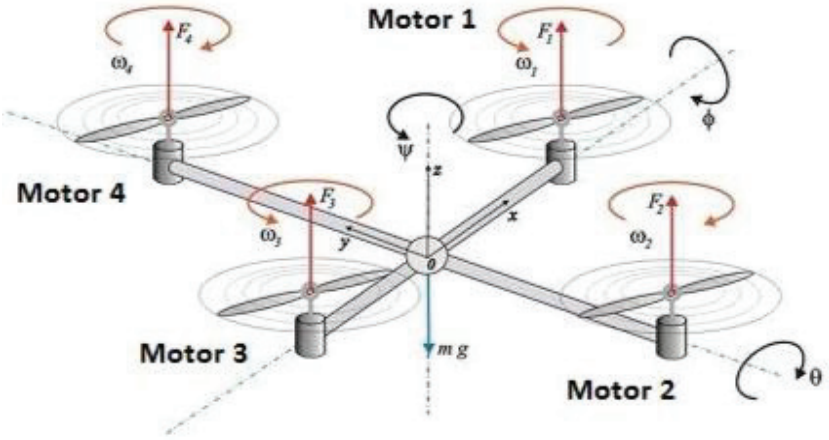
Fonte: Produzido pelos autores

A classificação dos *Vants* é vasta e com grandes variedades de tipos. Se destacam os *Vants* de reconhecimentos, de combate, aéreos, civil e comercial, de pesquisa e de mísseis de cruzeiro. A todos os tipos aplica-se a classificação de dados, locais inacessíveis, rápida aquisição, variando o tipo que é classificado. Cabe ressaltar que para ocorrer a operação de *VANT* em Espaço Aéreo, é necessário garantir níveis de segurança de aeronaves tripuladas.

4. QUADCOPTER

O Quadcopter é um veículo de asas rotativas, com 4 motores e hélices rotativas que produzem o equilíbrio do veículo.

Figura 5 - Quadcopter – Adaptador (VIKLUND, 2012)



Fonte: Produzido pelos autores

Com o *quadcopter* é possível locomoção em lugares de difíceis acesso, que necessitam de manobras e estabilidade é um veículo com capacidade de decolagem e aterrissagem em modo vertical, baseado nos seus 4 motores.

5. APLICAÇÕES

Os quadcopters são caracterizados como veículos não tripulados, são um tipo de *VANT*. Porém, autônomos e remotamente controláveis, ou seja, a presença de um humano controlando é inviável no processo. (ENGEL, 2012). Por serem controlados remotamente são aplicados em meios de vigilância, segurança, agricultura, a fim de proteger, garantir o processo com rapidez e precisão (JADHAV, 2013).

5.1. ARDUPILOT

Ardupilot é um repositório de conhecimento e inovação em constante evolução. O Ardupilot dividi-se em três partes: *software*, *firmware* e *hardware*. O *software* é a interface para o hardware, responsável pelas configurações iniciais, testes, missão de planejamento e análise pós-missão.

5.2. V-REP

O Virtual Robot Experimentation Plataform (V-REP) é um ambiente de desenvolvimento integrado, baseado em uma arquitetura de controle distribuída.

Cada modelo pode ser controlado individualmente através de um plugin, API,

ou uma solução personalizada. Sendo versátil para aplicações que possuem multi-robôs. Podendo ser desenvolvido em C/C++, Java, Phyton, Lua, Matlab, Urb ou Octave. Ideal para prototipagem rápida, simulação e monitoramento remoto.

O software será a interface para o hardware, a partir dele será possível simular o software de controle e gerenciar toda a interação com o hardware. Será planejada toda a operação de *waypoints* que tem como objetivo garantir um estado consistente entre o remetente e o receptor, sua origem e chegada. Assim, serão utilizados os points como referência, tendo sua configuração inicial, testes, missão de planejamento, operações e pós-operações para gerar todo o gerenciamento.

5.3. INTELIGÊNCIA ARTIFICIAL

Segundo Haykim, Simon (2001), o objetivo da Inteligência Artificial (IA)

> é o desenvolvimento de paradigmas ou algoritmos que requeiram máquinas para realizar tarefas cognitivas, para as quais os humanos são atualmente melhores [...]. Um sistema de IA deve ser capaz de fazer três coisas: (1) armazenar conhecimento, (2) aplicar o conhecimento armazenado para resolver problemas e (3) adquirir novo conhecimento através da experiência. Um sistema de IA tem três componentes fundamentais: representação, raciocínio e aprendizagem (HAYKIN, SIMON, 2001, p.59).

A teoria fundamentada por Haykin (2001) apresenta termos que devem ser adequados em alguns paradigmas, tendo que seguir um padrão criado. No qual um sistema de IA só é considerado quando: armazena conhecimento, sabendo utilizar a informação que é coletada, aplicar esse conhecimento armazenado e adquirir novo conhecimento.

5.4. LÓGICA FUZZY

A lógica nebulosa, Fuzzy Logic, permite o tratamento de expressões que envolvem grandezas de forma não exata (NASCIMENTO JÚNIOR, YONEYAMA, 2004, p. 68) e conceitos de universos diferentes que apresentam relações entre si.

A lógica fuzzy possui um conjunto, uma coleção de objetos, no qual um dado faz ou não faz parte do conjunto. A teoria dos conjuntos, na lógica fuzzy, demonstra a informação imprecisa que ocorre de maneira natural na linguagem humana, podendo ser aplicado em IA, na economia, processamento de imagens, dentre outros. (ALEXANDRE e NIVAL, 2015)

A lógica fuzzy possui conceitos relacionados a conjuntos fuzzy, controladores fuzzy, fuzzificador, inferência fuzzy, defuzzyficador e sua base de regras para controle do modelo de raciocínio.

5.5. LINGUAGEM

De acordo com a aplicação em desenvolvimento e escolhas de metodologia, foi escolhida a JFuzzyLogic, por ser implementada em linguagem Java e adaptável ao que será implementado posteriormente.

A JFuzzyLogic é um pacote de lógica fuzzy escrito em Java, uma *opensource* que implementa Fuzzy Controller Language (FCL). Uma linguagem robusta e flexível de sistema de inferência. Um FCL é um sistema de regras de lógica difusa baseada em conhecimento, regras de fuzzificação, inferência e defuzzificação.

O sistema de regras FCL é definido como linguagem de controle e baseado em entrada e saídas. São definidas as funções e variáveis para aplicação em lógica fuzzy. Seu funcionamento segue um padrão de FCLs (IEC 61131-7), padronizada pela IEC na categoria PLC (Controlador Lógica Programável). Podendo ter um conjunto de variáveis, operações que retorna um valor e um tipo primitivo de objeto. Um modelo básico de PLC é chamado de Unidade de Organização de Programas (POU) e cada um pode ser implementado de uma forma diferente (CAMPOS e RIBEIRO, 1994).

FIS, um sistema de inferência, é um tipo de POU e é composto por uma ou mais classes, tendo um antecedente, um consequente e um método que implica a regra.

Exemplos:

Regra: Se o serviço é ruim OU comida é rançoso ENTÃO dica é barato;

Antecedente: “Serviço é pobre OU alimento é rançoso”.

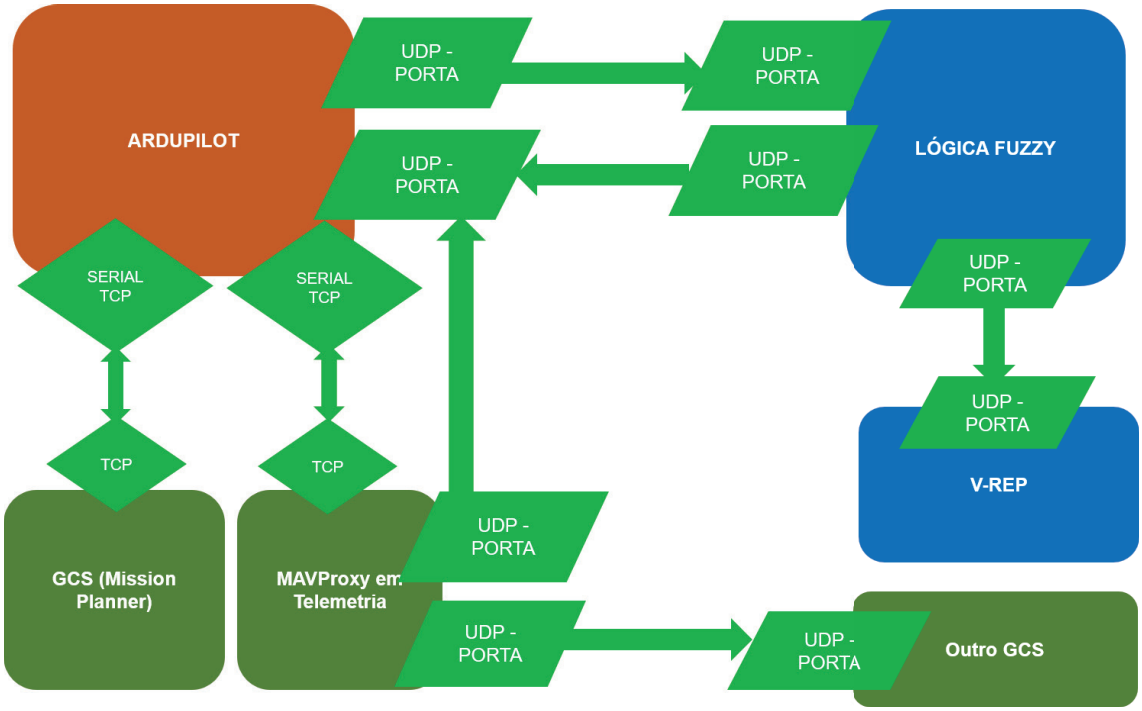
Consequência: “Ponta é barato”.

As consequências são uma coleção de classes RuleTerms. E um antecedente é representado por uma classe RuleExpression. O jFuzzyLogic fornece extensões convenientes para definir associações. São divididas em subclasses, sendo uma contínua e discreta. Fornece uma estrutura de otimização de parâmetros, podendo ser aplicados a conjuntos de dados para refinar os parâmetros de associação.

6. ARQUITETURA MODELO DA INTEGRAÇÃO ARDUPILOT/V-REP

A arquitetura modelo contempla indicativos de conexão entre interlocutores que podem variar. Portas entre o Ardupilot e o Simulador V-REP.

Figura 6 - Arquitetura Ardupilot/V-REP



Fonte: Produzido pelos autores

A arquitetura proposta é composta pelo Ardupilot um software de piloto automático de código aberto, capaz de controlar qualquer sistema de veículo imaginável, Multi Rotores e helicópteros compostos. O seu hardware pode ser instalado e otimizado. O Ardupilot varia com portas UDP, portas não orientadas para a conexão da camada de transporte do modelo TCP/IP. Pontes de comunicação que trocam comunicação entre protocolos, mas não os alteram

7. MAVLINK *VERSUS* SITL GAZEBO *VERSUS* ARDUPILOT/V-REP

O Mavlink, como arquitetura, aplica-se muito bem ao contexto desta aplicação, porém já é padrão um protocolo de comunicação de VANT programado em python, que de acordo com o modelo do veículo é necessária a realização de adaptações nos padrões de mensagem para comando de voo e vem com seu próprio simulador em telemetria.

Já o SITL GAZEBO permite executar todos os modelos dos plane, dando um executável nativo para testar qualquer código.

Porém, os dois, Mavlink e SITL GAZEBO, possuem seus simuladores embutidos e modelos que permitem executar o comportamento do código sem o hardware. A atual arquitetura apresentada e estudada é a sua junção, porém com melhorias.

É utilizado o V-REP como simulador de robôs, com ambiente de desenvolvimento integrado, com arquitetura de controle distribuído, tendo individualmente scripts, plug-ins, clientes API remoto e ROS que são versáteis, prototipação rápida, vários modelos e robôs embutidos, escritos e adaptável em várias linguagens.

O Ardupilot é o sistema de piloto automático de código aberto mais avançado e capaz de controlar qualquer veículo imaginável. Sendo capaz de passar qualquer tipo de informação de altitude e o local de decolagem para ter um melhor controle do veículo, que são todas as instruções que serão controladas pelo hardware. Tem o modelo lógico para controlar todo o fluxo de software e simulador. Esse modelo de raciocínio é o diferencial desta arquitetura. É uma técnica de inteligência artificial que permite expressões que envolvem grandezas descritas de forma não exata (NASCIEMNTO JÚNIOR, YONEAMA, 2004, p.68). São expressões próximas de ações das atitudes do ser humano. Ações naturais, mais adaptáveis à linguagem humana e não tanto a computacional.

Esse contexto ajuda muito no controle de um veículo, pois controlará como estão acostumados no dia a dia. Como: “ande mais um pouco para a esquerda”, por exemplo. É uma linguagem aceitável e de fácil adaptação, para controle de algo se torna tão comum, pois é algo rotineiro. Em outros termos, é uma arquitetura voltada para as necessidades humanas e demonstra mais facilmente a linguagem computacional.

8. DESENVOLVIMENTO

Para demonstrar o que foi proposto neste artigo, foi codificada uma simulação de voo controlada por lógica fuzzy, utilizando a linguagem Java, Remote API, V-REP e a biblioteca JFuzzyLogic. Esta simulação demonstra o uso de lógica fuzzy para controlar um voo simulado.

Para isso, foi necessário montar o ambiente no simulador V-REP com o quadcopter e os sensores accelerometer, gyroscope e proximity. Com o ambiente montado, passou-se à etapa da descrição da lógica fuzzy, na qual foram levantadas as variáveis, as grandezas e as regras para controle de voo. Após esta etapa, passou-se à codificação do algoritmo de conexão com o simulador V-REP utilizando remoteAPI e controle de voo com lógica fuzzy utilizando a biblioteca JFuzzyLogic.

Para realizar futuros testes, foi necessário criar a estrutura de como funcionaria a lógica fuzzy na prática. Foi preparada uma estrutura de variáveis de entrada, variáveis de saída e as regras para verificar a necessidade do problema e em seguida aplicá-lo.

Tabela 1 - Variáveis de entrada e saída com suas grandezas - Lógica Fuzzy

Variáveis de Entrada	**Grandezas**
Altura	Pousado
	Baixo
	Alto
Acelerometer X	Negativo
	Nivelado
	Positivo
Acelerometer Y	Negativo
	Nivelado
	Positivo
Variáveis de Saída	**Grandezas**
Motor 1	Baixa
	Estável
	Alta
Motor 2	Baixa
	Estável
	Alta
Motor 3	Baixa
	Estável
	Alta
Motor 4	Baixa
	Estável
	Alta

Fonte: autor.

Tabela 2 - Regras Fuzzy para controle do quadcopter

Regras Fuzzy

Se Altura == Pousado && AccX == nivelado && AccY == nivelado -> (M1, M2, M3, M4) = alta
Se Altura == Baixo && AccX == nivelado && AccY == nivelado -> (M1, M2, M3, M4) = alta
Se Altura == Alto && AccX == nivelado && AccY == nivelado -> (M1, M2, M3, M4) = estável
Se Altura != alto && AccX == negativo -> (M1, M4) = estável
Se Altura == alto && AccX == negativo -> (M1, M4) = baixo
Se Altura != alto && AccX == positivo -> (M2, M3) = estável
Se Altura == alto && AccX == positivo -> (M2, M3) = baixo
Se Altura != alto && AccY == negativo -> (M1, M2) = estável
Se Altura == alto && AccY == negativo -> (M1, M2) = baixo
Se Altura != alto && AccY == positivo -> (M3, M4) = estável
Se Altura == alto && AccY == positivo -> (M3, M4) = baixo

Fonte: autor.

Na tabela 1 e 2, é relatada a estrutura principal do desenvolvimento deste artigo. São classificadas as entradas que foram utilizadas, cada entrada possui sua grandeza, ou seja, um grau que pode alcançar no decorrer da sua execução. No mesmo caso, para as variáveis de saída.

Além das regras fuzzy, a partir das entradas podem ser ativadas para uma saída esperada.

O ponto relevante é que são variáveis que ajudam a atingir o voo estável do componente utilizado, o quadcopter. De acordo com as grandezas, a base para as regras utilizadas é a variável “altura”, toda a base de voo do quadcopter se baseia nela. A partir da altura, é possível delimitar o tipo do voo do componente quadcopter.

A partir disso, o quadcopter deverá ficar numa altura estável, quanto mais baixo, mais velocidade é aplicada aos motores, até atingir uma altura adequada. Mesmo que se aplique variáveis positivas e negativas aos motores para controlar a altura do voo para estável. Como as regras demonstram, em diferentes pontos no acelerometer, estando com inclinação positiva ou não, a velocidade aplicada será proporcional ao motor que está com a variável deixando o voo instável, ou seja, o quadcopter vai estar estável nos 4 motores, nos dois eixos do acelerometer e sempre aplicando a maior velocidade nos motores necessários para estabilizar o voo. Com esse contexto estruturado, foi possível realizar alguns testes.

Tendo em vista as regras, o algoritmo utilizou a altura capturada pelo sensor de proximidade para fazer o quadcopter levantar voo até uma determinada altura e estabilizá-lo. Portanto, utilizou as informações capturadas pelo accelerometer para estabilizar o voo do quadcopter.

9. CONCLUSÃO

Analisando o desenvolvimento obtido ficou claro que a implementação da integração da arquitetura Ardupilot com o V-REP obteve sucesso. Comparando os resultados, tem-se limitações no decorrer do desenvolvimento no qual não foi possível introduzir a implementação dos sensores accelerometer e gyroscopio para testes, porém, foram feitos testes iniciais com altura para que o quadcopter atingisse e pudesse chegar a uma estabilidade por meio de integração com regras fuzzy. Foi um teste necessário, pois preparou o ambiente para receber os sensores. O controle da altura foi alcançado, não de forma ideal, com consequências de instabilidade de percurso e voo sem altura definida.

Uma solução é estender os testes para aplicação dos sensores com as regras fuzzy, procurar fazer a intercalação para se comunicarem.

Com o desenvolvimento desta pesquisa, propõe-se, posteriormente, a utilização da lógica fuzzy e redes neurais. Um caminho possível para controle autônomo do quadcopter, no qual será realizado o seu controle por aprendizagem RNA e Fuzzy, obtendo controle autônomo do quadcopter e capacidade de aprendizagem autônoma.

REFERÊNCIAS

Alexandre G., Nival N. **Integração de um sistema de inferência fuzzy em ambiente de simulação robótica para aprendizagem de técnicas inteligentes.** XII Simpósio Brasileiro de Automação Inteligente (SBAI). Natal - RN, 25 a 28 de out. 2015.

ARDUPILOT DEV TEAM. **ArduPilot.** 2016. Disponível em: < http://ardupilot.com/ardupilot/index.html >. Acesso em: 19 ago. 2016.

BEHNCK, Lucas Pluceno. **Controle de Missão de Voo de Veículo Aéreo Não-Tripulado**. Porto Alegre. 2014.

BRAGA, Alex Pereira. **Simulação e controle de um quadcopter utilizando lógica fuzzy.** Trabalho de Graduação em Engenharia Elétrica – Universidade Estadual Paulista, Guaratinguetá: [s.n], 2013 89 fls.

CAMPOS, Gomide; RIBEIRO, Gudwin. **Modelagem, Controle, Sistemas e Lógica Fuzzy** – São Paulo. Universidade Estadual de Campinas (UNICAMP). SBA Controle & Automação. Vol.4. Set – Out. 1994.

MORAIS, Bruno Santos; SILVA, Ádria Aline Castro da; OLIVEIRA, Omar Kayque Camargo; Igor Yepes. **Protótipo de Veículo Aéreo Não Tripulado para Monitoramento Ambiental e Apoio a Missões USAR**. Computer on The Beach 2014, At Florianópolis/SC, 2014.

SILVA, Cláudia R. de S. e; YEPES, Igor. **Desenvolvimento de sistema SLAM com odometria visual para VANT de inspeção em ambientes interno**. Palmas, Tocantins. Abril,2015.

RÊGO, Cristian D. A.; YEPES, Igor. **Sistema de controle de VANT mediante gestos utilizando o sensor de movimentos KINECT**. Palmas, Tocantins. Maio,2015.

NASCIMENTO JÚNIOR, Cairo Lúcio; YONEYAMA, Takashi **Inteligência artificial em controle e automação**. São Paulo: Blucher, Fapesp, 2004.

APÊNDICE

Figura 7 - Cena quadcopter estático – V-REP

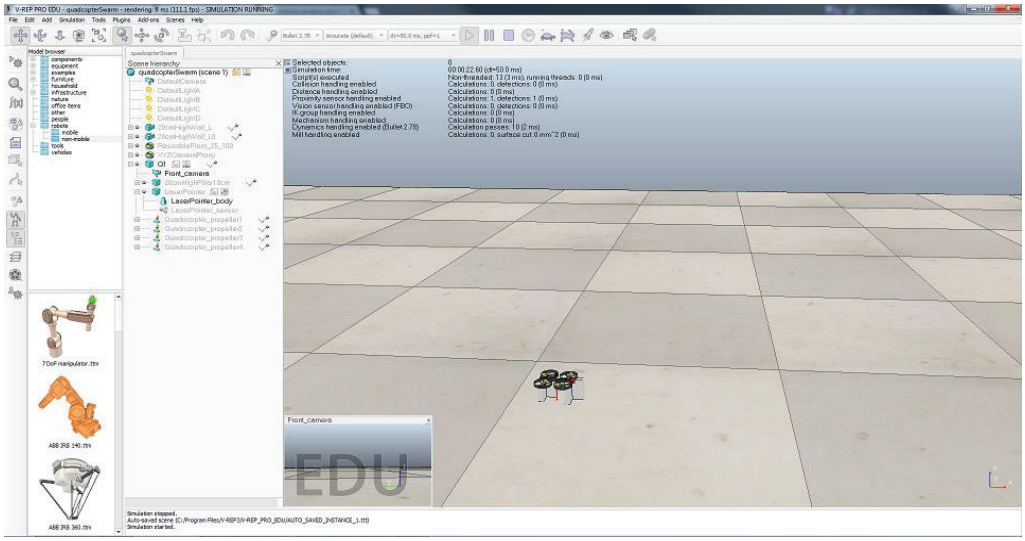


Figura 8 - Cena quadcopter voando – V-REP