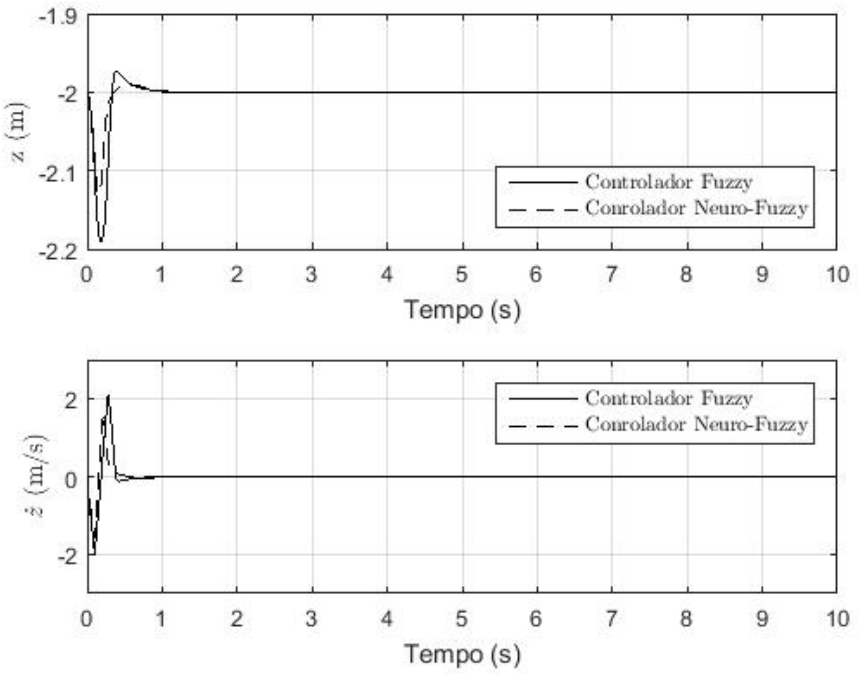
1 Resultados

A Figura 1 mostra a posição no eixo vertical (z) do *drone*, bem como sua variação no sistema em que atua o controlador *fuzzy* projetado. Como se pode ver, o distúrbio foi devidamente controlado, fazendo com que o quadricóptero retornasse à posição inicial $z = -2$ m e também ao repouso[1] representado por $\dot{z} = 0$ m/s. Nesta figura, entretanto, não fica tão clara a diferença de desempenho dos controladores *fuzzy* e neuro-*fuzzy*, aspecto que pode ser claramente verificado na Figura 2. Como se pode ver, tanto para z quanto para $\dot{z}$, o neuro-*fuzzy* apresenta desempenho melhor. No controle sobre a posição z, o controlador neuro-*fuzzy* apresentou redução do tempo de convergência em 29%, e da variação do sistema em 31%, além de eliminar a sobrelevação apresentada pelo *fuzzy*. Já sobre a velocidade $\dot{z}$, apresentou uma redução no tempo de convergência de 29% além melhorar a variação do sistema em 23%. A partir destes resultados, verifica-se que o controlador neuro-*fuzzy* fez com que o distúrbio fosse melhor absorvido e que sua correção ocorresse mais rapidamente.

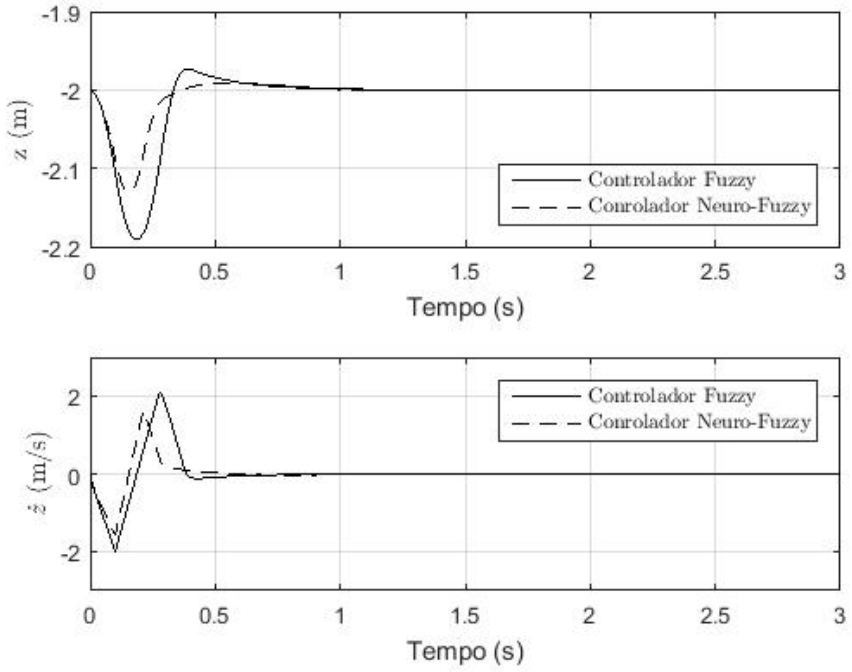
Figura 1 – Comparação da resposta das saídas z e $\dot{z}$ no controle de altitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 2$ kg



Já as Figuras 3 e 4 mostram a estabilidade de atitude em torno dos eixos x e y (i.e em relação ao plano horizontal XY), representados por θ e ϕ respectivamente. Como se pode ver, ambos os estados são devidamente controlados e, com isto, o *drone*

[1] i.e. velocidade nula

Figura 2 – Comparação em mais detalhes da resposta das saídas z e $\dot{z}$ no controle de altitude fuzzy e neuro-fuzzy para o sistema com massa $m = 2$ kg



volta à estabilidade horizontal, com ângulos e velocidades angulares nulas no estado permanente.

Figura 3 – Comparação da resposta das saídas ϕ e $\dot{\phi}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 2$ kg

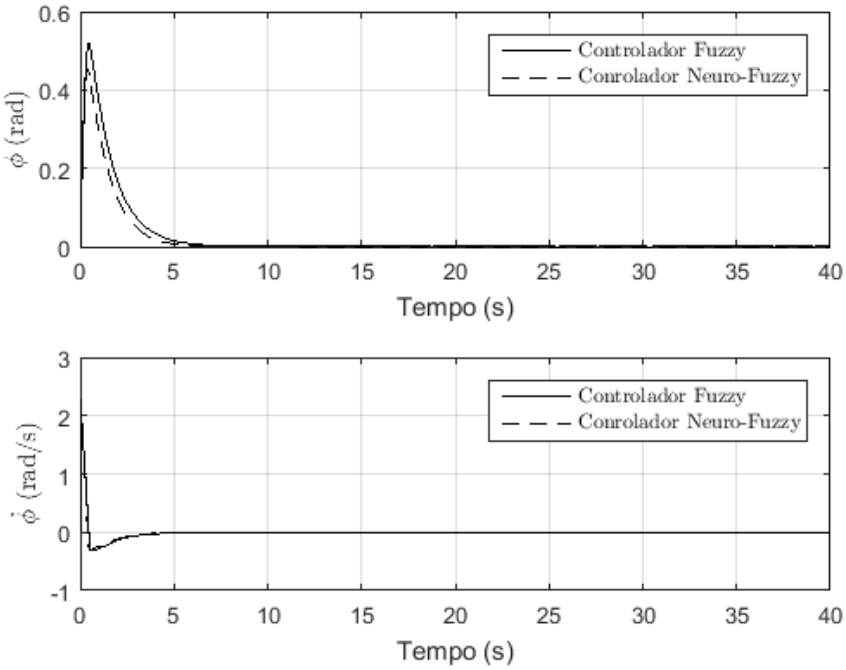
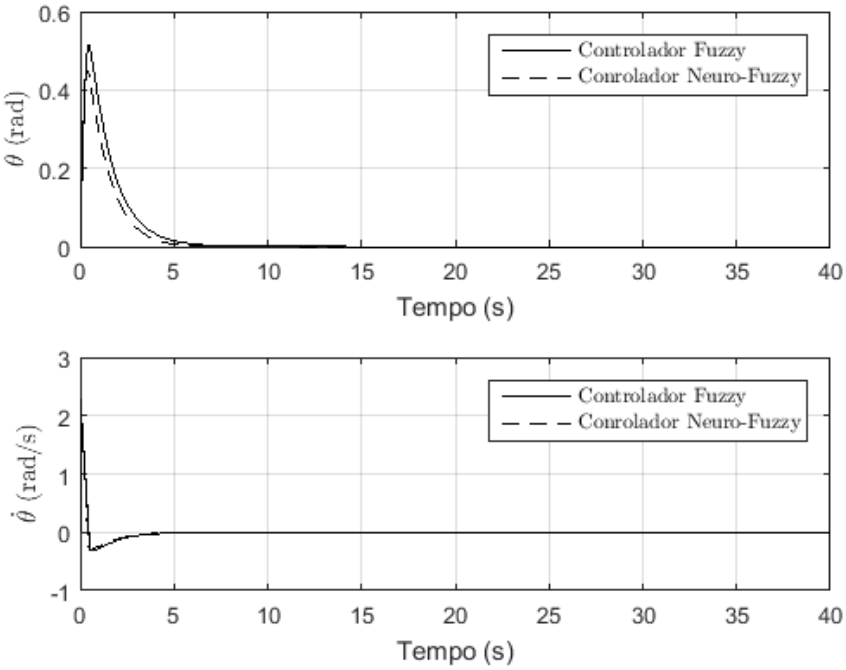


Figura 4 – Comparação da resposta das saídas θ e $\dot{\theta}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 2$ kg



A partir das Figuras 5 e 6, que mostram as respostas obtidas em mais detalhes, pode-se ver que, mais uma vez o controlador neuro-*fuzzy* mais uma vez teve desempenho superior ao *fuzzy*, fazendo com que os ângulos ϕ e θ convergissem 2% mais rápido, além de reduzir suas variações a 13%. Com relação às velocidades angulares $\dot{\phi}$ e $\dot{\theta}$, foi capaz de reduzir o tempo de convergência em 3%, não afetando a variação nem a sobrelevação apresentada pelo sistema quando estabilizado pelo controlador *fuzzy*. Desta forma, verifica-se que o controle neuro-fuzzy levou o sistema a uma menor variação, representando que o ângulo máximo de inclinação alcançado pelo *drone* é menor e corrigido mais rapidamente.

Além da verificação da eficiência dos controladores atuando sobre o sistema para o qual foram projetados, eles foram testados num sistema em que um dos parâmetros foi acrescido de mais de 100 %, com a massa passando de 2,3 kg para 5 kg.

A resposta dos controladores *fuzzy* e neuro-*fuzzy* para altitude do *drone* nessas circunstâncias são mostradas nas Figuras 7 e 8, sendo que esta segunda é apenas uma forma melhor de comparar a ação dos dois controladores. Como se pode perceber, ambos os controladores levaram à estabilização do sistema, sendo que desta vez cada um obteve desempenho melhor sob determinados aspectos. No controle da posição vertical z, o neuro-fuzzy apresentou tempo de convergência 57% maior, em parte causado por uma sobrelevação, que não foi apresentada pelo *fuzzy*. Em contrapartida, o neuro-fuzzy apresentou uma menor variação, a reduzindo em 20% se comparado ao *fuzzy*. Com

Figura 5 – Comparação em mais detalhes da resposta das saídas ϕ e $\dot{\phi}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 2$ kg

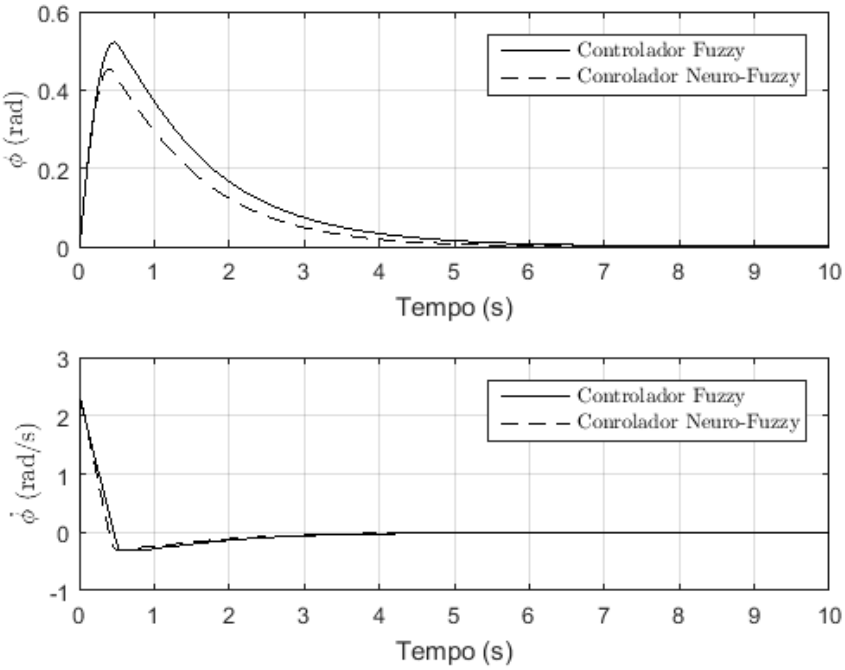
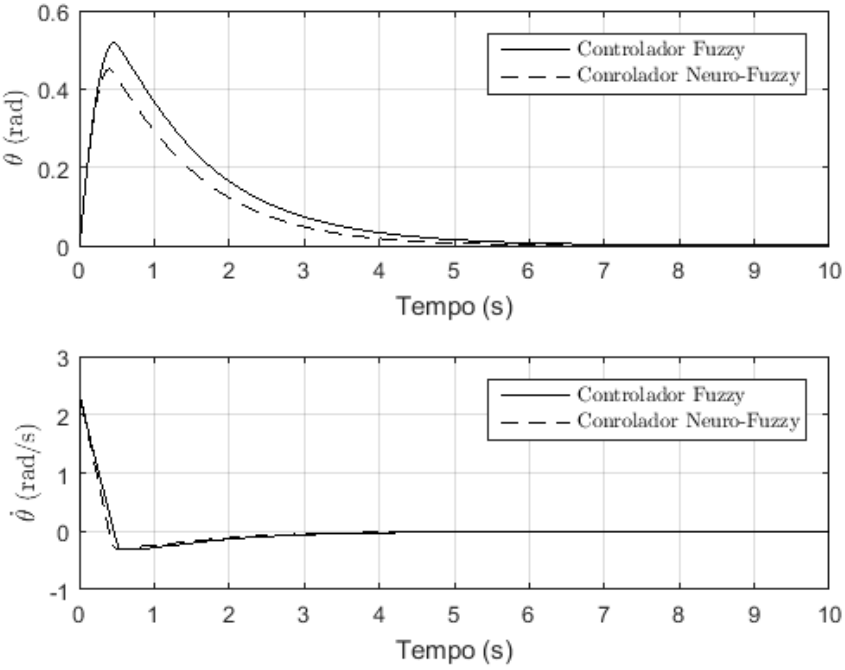


Figura 6 – Comparação em mais detalhes da resposta das saídas θ e $\dot{\theta}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 2$ kg



relação à velocidade $\dot{z}$, o controlador neuro-*fuzzy* apresentou aumento de 23% no tempo de convergência, mas melhorou o sistema nos quesitos variação e sobrelevação, as reduzindo em 16% e 33%, respectivamente. Esses resultados apontam que o quadricóptero,

quando submetido ao controle neuro-*fuzzy*, apresentou movimentos mais suaves até ter sua altitude estabilizada, apesar de ter sido necessário mais tempo para que ela ocorresse.

Figura 7 – Comparação da resposta das saídas z e $\dot{z}$ no controle de altitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 5$ kg

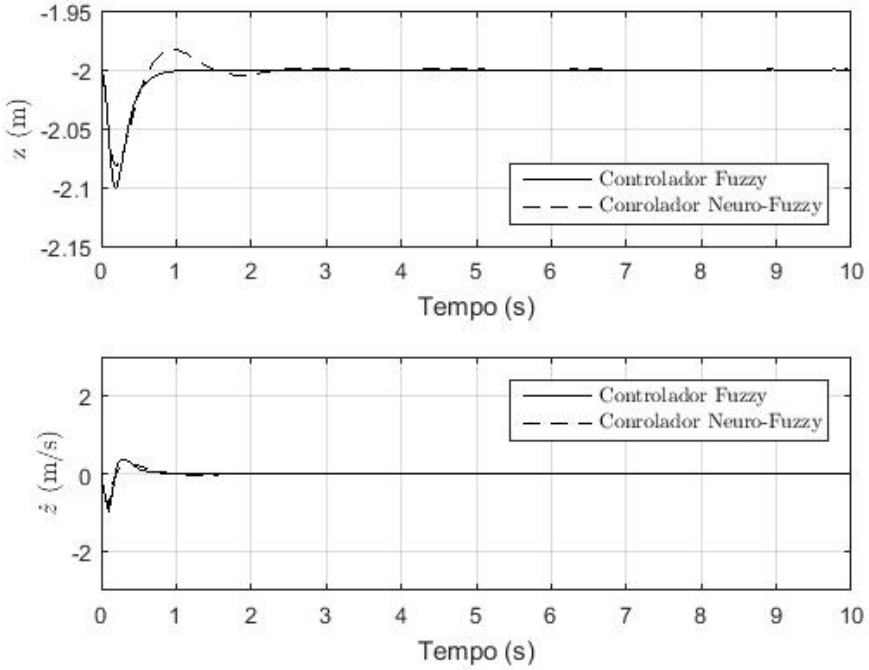
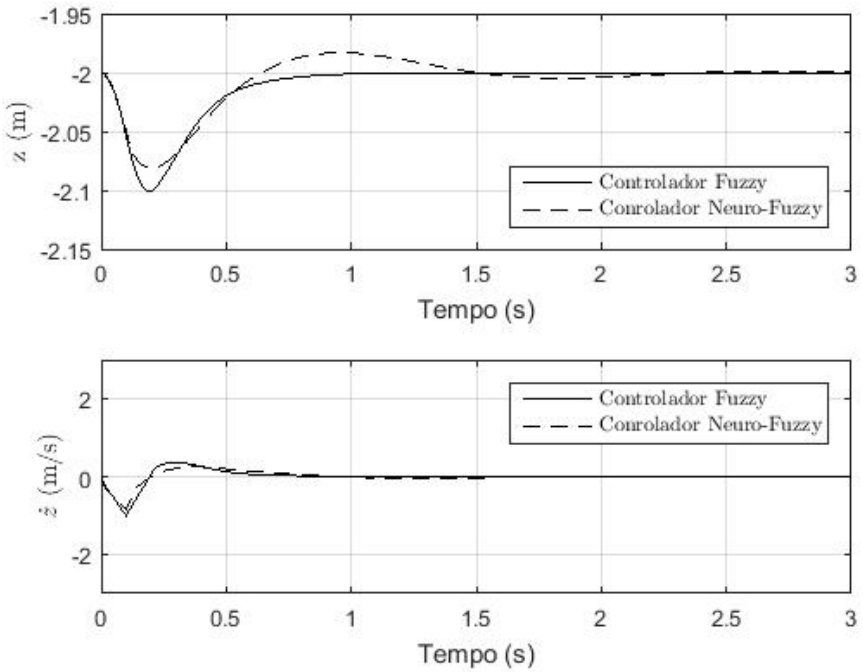


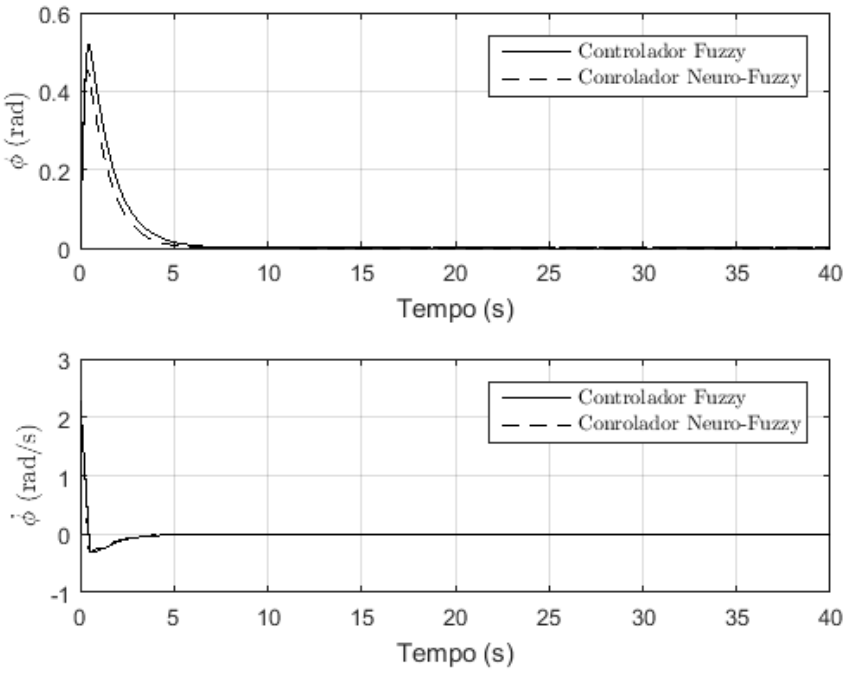
Figura 8 – Comparação em mais detalhes da resposta das saídas z e $\dot{z}$ no controle de altitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 5$ kg



Por fim, as Figuras 9 e 10 mostram os resultados obtidos pelos controladores de atitude no sistema com massa $m = 5$ kg. Percebe-se que mais uma vez o sistema convergiu ao seu estado de estabilidade com os ângulos nulos e velocidades angulares também nulas, representando que o quadricóptero, após a ação de controle, tanto *fuzzy* quanto neuro-*fuzzy*, fica estável e com orientação plana[2].

As Figuras 11 e 12 mostram em mais detalhes as repostas obtidas pelos controladores sobre a atitude do sistema com massa $m = 5$ kg. A partir delas, nota-se que mais uma vez o controlador neuro-*fuzzy* apresentou desempenho levemente superior ao *fuzzy*. Sobre os ângulos ϕ e θ, a convergência ocorreu 3% mais rapidamente e a variação apresentada reduziu 14%. Já sobre as velocidades angulares $\dot{\phi}$ e $\dot{\theta}$, a redução de tempo de convergência com o neuro-*fuzzy* foi de 2%, mantendo a mesma sobrelevação e variação oferecidas pelo *fuzzy*. Com isto, mais uma vez o controle neuro-fuzzy fez com que o ângulo máximo de inclinação do *drone* fosse inferior ao alcançado pelo sistema controlado pelo *fuzzy*,e além de reduzir o tempo necessário para sua estabilização definitiva.

Figura 9 – Comparação da resposta das saídas ϕ e $\dot{\phi}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 5$ kg



1.1 Análise Consolidada dos Resultados

Como se pode ver, os controladores de atitude e altitude tanto fuzzy quanto neuro-fuzzy foram eficientes levando à estabilização do sistema em todos os casos

[2] i.e paralela ao plano XY

Figura 10 – Comparação da resposta das saídas θ e $\dot{\theta}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 5$ kg

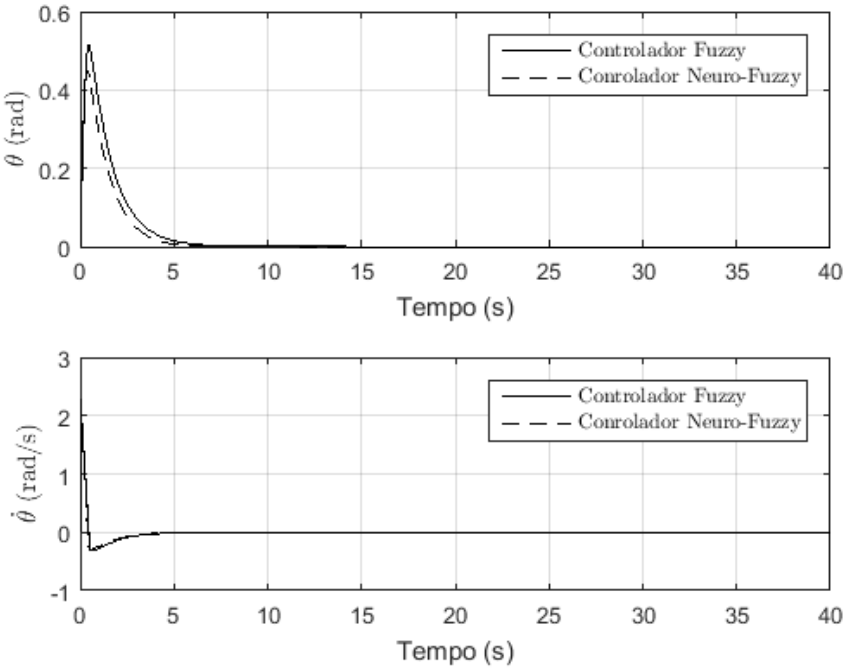
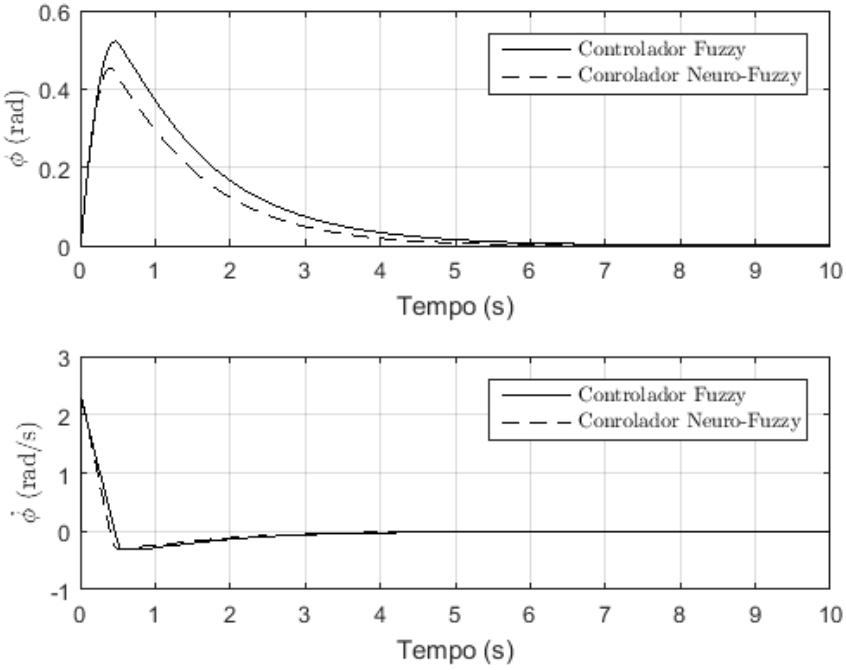
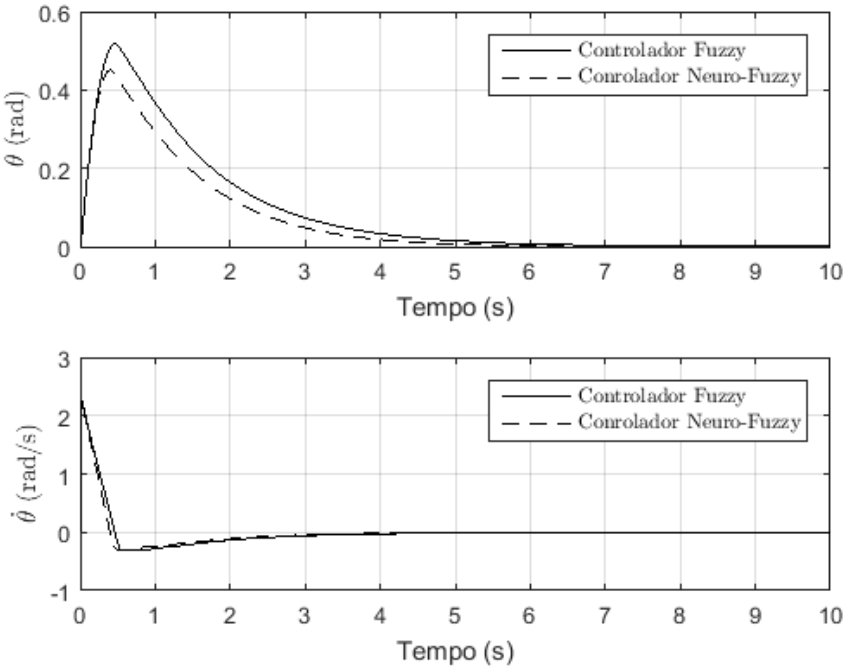


Figura 11 – Comparação da resposta das saídas ϕ e $\dot{\phi}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 5$ kg



testados, inclusive na situação em que a massa do sistema foi aumentada em mais de 100 %. Isso indica que estes controladores podem ser utilizados, por exemplo, em situações em que o drone precisaria transportar uma carga que tenha sua massa ou até

Figura 12 – Comparação da resposta das saídas θ e $\dot{\theta}$ no controle de atitude *fuzzy* e neuro-*fuzzy* para o sistema com massa $m = 5$ kg



mesmo uma superior.

Em quase todos os casos, nota-se um comportamento do controlador neuro-fuzzy superior ao do fuzzy, o que já era esperado tendo em vista que o primeiro alia o poder do segundo às vantagens das RNAs, fazendo com que, a partir de um treinamento supervisionado utilizando o próprio modelo fuzzy, possa se construir um controle mais abrangente e com resposta melhorada.

Estes resultados mostram que, de fato, as técnicas de Inteligência Computacional podem ser aplicadas para projetar controladores eficientes e robustos para atuar sobre sistemas multivariável e que, além disto, o poder de treinamento dos ANFISs realmente é capaz de fazer com que o desempenho de controladores neuro-fuzzy seja melhorado se comparado a de controladores puramente fuzzy.

Referências

Apêndices

APÊNDICE A – Código para Criação de Modelo Neuro-Fuzzy para Altitude e Definição de Dados para Treinamento

```
% le arquivo fis referente ao controle de altitude
fismat = readfis('fis_altitude.fis');

% define numero de casos a serem avaliados (treinamento + teste)
n = 300;
% define conjunto de n entradas aleatorias para o sistema fuzzy
% respeitando o range de cada entrada
input = zeros(n,2);
for i=1:n
    z_value = rand * 2 - 1;
    z_dot_value = rand * 10 - 5;
    input(i,:) = [ z_value z_dot_value ];
end

% avalia resposta fuzzy para cada entrada
output= evalfis(input,fismat);

% define data como vetor relacionando cada conjunto de entradas ...
   a saida
% - obtida pelo sistema fuzzy
data = [];
for i=1:n
   data(i,:) = [ input(i,:) output(i) ];
end

% define que 2/3 dos dados obtidos serao usasdos para treinamento
% e 1/3 sera usado para teste da rede
train = data(1:2*n/3,:);     % dados para treinamento
test = data(2*n/3+1:n,:);    % dados para validacao do sistema ...
   treinado

% gera modelo fuzzy Sugeno a partir do Mamdani modelado
sugFIS = mam2sug(fismat);
% salva modelo Sugeno em disco com o nome fis_altitude_neuro.fis
writefis(sugFIS, 'fis_altitude_neuro.fis');
```

APÊNDICE B – Código para Criação de Modelo Neuro-Fuzzy para Atitude e Definição de Dados para Treinamento

```
% le arquivo fis referente ao controle de atitude
fismat = readfis('fis_atitude.fis');

% define numero de casos a serem avaliados (treinamento + teste)
n = 300;
% define conjunto de n entradas aleatorias para o sistema fuzzy
% respeitando o range de cada entrada
input = zeros(n,2);
for i=1:n
    phi_value = rand * 4 - 2;
    phi_dot_value = rand * 3 - 1.5;
    input(i,:) = [ phi_value phi_dot_value ];
end

% avalia resposta fuzzy para cada entrada
output= evalfis(input,fismat);

% define data como vetor relacionando cada conjunto de entradas ...
   a saÃda
% obtida pelo sistema fuzzy
data = [];
for i=1:n
   data(i,:) = [ input(i,:) output(i) ];
end

% define que 2/3 dos dados obtidos serao usasdos para treinamento
% e 1/3 sera usado para teste da rede
train = data(1:2*n/3,:);     % dados para treinamento
test = data(2*n/3+1:n,:);    % dados para validaÃ§Ã£o do sistema ...
   treinado

% gera modelo fuzzy Sugeno a partir do Mamdani modelado
sugFIS = mam2sug(fismat);
% salva modelo Sugeno em disco com o nome fis_atitude_neuro.fis
writefis(sugFIS, 'fis_atitude_neuro.fis');
```