**Druhá průběžná práce NNUI1**

Navrhněte a implementujte fuzzy inferenční systém typu Mamdani, který určuje zrychlení autonomního vozidla na základě aktuálních dopravních podmínek.

Podmínky úlohy:

1) Autonomní vozidlo musí reagovat na změny dopravních podmínek, aby zajistilo plynulou a bezpečnou jízdu.

2) Parametry vstupující do systému:

a. Vzdálenost od předchozího vozidla či překážky před vozidlem (0 až 120 m)

b. Relativní rychlost vůči předchozímu vozidlu či překážce před vozidlem (-100 až 100 km/h).

c. Intenzita dopravy (míra hustoty vozidel v okolí) na škále 0 až 10, kde 0 představuje minimum aut v okolí, možnost snadného předjíždění nebo udržení bezpečné vzdálenosti a 10 znamená vysokou hustotu s minimálními mezerami mezi vozidly.

3) Na základě těchto parametrů je nutné určit výstup představující zrychlení vozidla na škále -5 až 5 .

V rámci řešení navrhněte řešení v libovolném programovacím jazyku. Zdůvodněte tvorbu pravidel, implementujte fuzzy inferenční systém a proveďte evaluaci minimálně pro 5 vybraných kombinací vstupních parametru. Výslednou hodnotu zrychlení poskytnutou navrženým fuzzy inferenčním systémem diskutujte.

Odevzdaná práce musí obsahovat:

• Report o řešení v rozsahu textu výše

• Kompletní zdrojové kódy aplikace

• Spustitelná aplikace s kompletním návodem, jak aplikaci oživit na libovolném počítači (Windows 10, 11 nebo Linux Ubuntu 20.04 LTS)

Táto úloha navrhnuta v Pythonu.

Zdrojový kód

Fuzzy\_system.py

**import** numpy **as** np

**class** **FuzzySet:**

**def** \_\_init\_\_**(**self**,** name**,** membership\_function**):**

self**.**name **=** name

self**.**membership\_function **=** membership\_function

**def** membership**(**self**,** x**):**

**return** self**.**membership\_function**(**x**)**

**class** **LingVariable:**

**def** \_\_init\_\_**(**self**,** name**):**

self**.**name **=** name

self**.**sets **=** **{}**

**def** add\_set**(**self**,** fuzzy\_set**):**

self**.**sets**[**fuzzy\_set**.**name**]** **=** fuzzy\_set

**def** fuzzify**(**self**,** value**):**

memberships **=** **{}**

**for** name**,** fuzzy\_set **in** self**.**sets**.**items**():**

memberships**[**name**]** **=** fuzzy\_set**.**membership**(**value**)**

**return** memberships

**class** **FuzzyRule:**

**def** \_\_init\_\_**(**self**,** antecedents**,** consequent**):**

self**.**antecedents **=** antecedents

self**.**consequent **=** consequent

**def** evaluate**(**self**,** fuzzified\_inputs**):**

memberships **=** **[**fuzzified\_inputs**[**var**][**fuzzy\_set**]** **for** var**,** fuzzy\_set **in** self**.**antecedents**]**

**return** **min(**memberships**)**

**class** **FuzzySystem:**

**def** \_\_init\_\_**(**self**):**

self**.**variables **=** **{}**

self**.**rules **=** **[]**

self**.**output\_variable **=** **None**

**def** add\_variable**(**self**,** fuzzy\_variable**,** is\_output**=False):**

**if** is\_output**:**

self**.**output\_variable **=** fuzzy\_variable

**else:**

self**.**variables**[**fuzzy\_variable**.**name**]** **=** fuzzy\_variable

**def** add\_rule**(**self**,** rule**):**

self**.**rules**.**append**(**rule**)**

**def** infer**(**self**,** inputs**):**

fuzzified\_inputs **=** **{**name**:** var**.**fuzzify**(**inputs**[**name**])** **for** name**,** var **in** self**.**variables**.**items**()}**

output\_sets **=** **{**name**:** **[]** **for** name **in** self**.**output\_variable**.**sets**}**

**for** rule **in** self**.**rules**:**

activation **=** rule**.**evaluate**(**fuzzified\_inputs**)**

output\_set\_name **=** rule**.**consequent**[**1**]**

output\_sets**[**output\_set\_name**].**append**(**activation**)**

aggregated\_output **=** **{**name**:** **max(**values**)** **if** values **else** 0 **for** name**,** values **in** output\_sets**.**items**()}**

**return** aggregated\_output

**def** defuzzify**(**self**,** aggregated\_output**,** resolution**=**1000**):**

x **=** np**.**linspace**(**100**,** 0**,** resolution**)**

max\_xi **=** 0

max\_membership **=** 0

**for** xi **in** x**:**

**for** set\_name**,** membership\_value **in** aggregated\_output**.**items**():**

membership\_value\_temp **=** **min(**self**.**output\_variable**.**sets**[**set\_name**].**membership**(**xi**),** membership\_value**)**

**if** membership\_value\_temp **>** max\_membership**:**

max\_membership **=** membership\_value\_temp

max\_xi **=** xi

**return** max\_xi

**def** triangle**(**x**,** a**,** b**,** c**):**

**if** x **<=** a **or** x **>=** c**:**

**return** 0

**elif** a **<** x **<** b**:**

**return** **(**x **-** a**)** **/** **(**b **-** a**)**

**else:**

**return** **(**c **-** x**)** **/** **(**c **-** b**)**

main.py

**from** Fuzzy\_system **import** FuzzySet**,** LingVariable**,** FuzzyRule**,** FuzzySystem**,** triangle

# Definice vstupních fuzzy proměnných

vzdalenost **=** LingVariable**(**'Vzdalenost'**)**

vzdalenost**.**add\_set**(**FuzzySet**(**'Blizko'**,** **lambda** x**:** triangle**(**x**,** 0**,** 0**,** 40**)))**

vzdalenost**.**add\_set**(**FuzzySet**(**'Stredne'**,** **lambda** x**:** triangle**(**x**,** 20**,** 60**,** 100**)))**

vzdalenost**.**add\_set**(**FuzzySet**(**'Daleko'**,** **lambda** x**:** triangle**(**x**,** 80**,** 120**,** 120**)))**

relativni\_rychlost **=** LingVariable**(**'RelativniRychlost'**)**

relativni\_rychlost**.**add\_set**(**FuzzySet**(**'VelmiPomale'**,** **lambda** x**:** triangle**(**x**,** **-**100**,** **-**100**,** **-**50**)))**

relativni\_rychlost**.**add\_set**(**FuzzySet**(**'Pomale'**,** **lambda** x**:** triangle**(**x**,** **-**75**,** **-**25**,** 0**)))**

relativni\_rychlost**.**add\_set**(**FuzzySet**(**'Stejne'**,** **lambda** x**:** triangle**(**x**,** **-**10**,** 0**,** 10**)))**

relativni\_rychlost**.**add\_set**(**FuzzySet**(**'Rychle'**,** **lambda** x**:** triangle**(**x**,** 0**,** 25**,** 75**)))**

relativni\_rychlost**.**add\_set**(**FuzzySet**(**'VelmiRychle'**,** **lambda** x**:** triangle**(**x**,** 50**,** 100**,** 100**)))**

intenzita\_dopravy **=** LingVariable**(**'IntenzitaDopravy'**)**

intenzita\_dopravy**.**add\_set**(**FuzzySet**(**'Nizka'**,** **lambda** x**:** triangle**(**x**,** 0**,** 0**,** 3**)))**

intenzita\_dopravy**.**add\_set**(**FuzzySet**(**'Stredni'**,** **lambda** x**:** triangle**(**x**,** 2**,** 5**,** 8**)))**

intenzita\_dopravy**.**add\_set**(**FuzzySet**(**'Vysoka'**,** **lambda** x**:** triangle**(**x**,** 7**,** 10**,** 10**)))**

# Definice výstupní fuzzy proměnné

zrychleni **=** LingVariable**(**'Zrychleni'**)**

zrychleni**.**add\_set**(**FuzzySet**(**'Brzdeni'**,** **lambda** x**:** triangle**(**x**,** **-**5**,** **-**5**,** **-**2.5**)))**

zrychleni**.**add\_set**(**FuzzySet**(**'Zadne'**,** **lambda** x**:** triangle**(**x**,** **-**2.5**,** 0**,** 2.5**)))**

zrychleni**.**add\_set**(**FuzzySet**(**'Akcelerace'**,** **lambda** x**:** triangle**(**x**,** 2.5**,** 5**,** 5**)))**

# Pravidla fuzzy systému

rules **=** **[**

# Pokud je vzdálenost blízko, relativní rychlost velmi pomalá a intenzita dopravy vysoká -> brzdit

FuzzyRule**([(**'Vzdalenost'**,** 'Blizko'**),** **(**'RelativniRychlost'**,** 'VelmiPomale'**),** **(**'IntenzitaDopravy'**,** 'Vysoka'**)],** **(**'Zrychleni'**,** 'Brzdeni'**)),**

# Pokud je vzdálenost blízko, relativní rychlost stejná a intenzita dopravy střední -> brzdit

FuzzyRule**([(**'Vzdalenost'**,** 'Blizko'**),** **(**'RelativniRychlost'**,** 'Stejne'**),** **(**'IntenzitaDopravy'**,** 'Stredni'**)],** **(**'Zrychleni'**,** 'Brzdeni'**)),**

# Pokud je vzdálenost střední, relativní rychlost pomalá a intenzita dopravy nízká -> žádné zrychlení

FuzzyRule**([(**'Vzdalenost'**,** 'Stredne'**),** **(**'RelativniRychlost'**,** 'Pomale'**),** **(**'IntenzitaDopravy'**,** 'Nizka'**)],** **(**'Zrychleni'**,** 'Zadne'**)),**

# Pokud je vzdálenost daleko, relativní rychlost rychlá a intenzita dopravy střední -> akcelerovat

FuzzyRule**([(**'Vzdalenost'**,** 'Daleko'**),** **(**'RelativniRychlost'**,** 'Rychle'**),** **(**'IntenzitaDopravy'**,** 'Stredni'**)],** **(**'Zrychleni'**,** 'Akcelerace'**)),**

# Pokud je vzdálenost daleko, relativní rychlost velmi rychlá a intenzita dopravy nízká -> akcelerovat

FuzzyRule**([(**'Vzdalenost'**,** 'Daleko'**),** **(**'RelativniRychlost'**,** 'VelmiRychle'**),** **(**'IntenzitaDopravy'**,** 'Nizka'**)],** **(**'Zrychleni'**,** 'Akcelerace'**))**

**]**

# Vytvoření fuzzy systému

fuzzy\_system **=** FuzzySystem**()**

fuzzy\_system**.**add\_variable**(**vzdalenost**)**

fuzzy\_system**.**add\_variable**(**relativni\_rychlost**)**

fuzzy\_system**.**add\_variable**(**intenzita\_dopravy**)**

fuzzy\_system**.**add\_variable**(**zrychleni**,** is\_output**=True)**

**for** rule **in** rules**:**

fuzzy\_system**.**add\_rule**(**rule**)**

# Testování systému pro různé kombinace vstupů

test\_inputs **=** **[**

**{**'Vzdalenost'**:** 10**,** 'RelativniRychlost'**:** **-**90**,** 'IntenzitaDopravy'**:** 9**},**

**{**'Vzdalenost'**:** 50**,** 'RelativniRychlost'**:** 0**,** 'IntenzitaDopravy'**:** 5**},**

**{**'Vzdalenost'**:** 100**,** 'RelativniRychlost'**:** 60**,** 'IntenzitaDopravy'**:** 2**},**

**{**'Vzdalenost'**:** 30**,** 'RelativniRychlost'**:** **-**20**,** 'IntenzitaDopravy'**:** 7**},**

**{**'Vzdalenost'**:** 80**,** 'RelativniRychlost'**:** 90**,** 'IntenzitaDopravy'**:** 1**}**

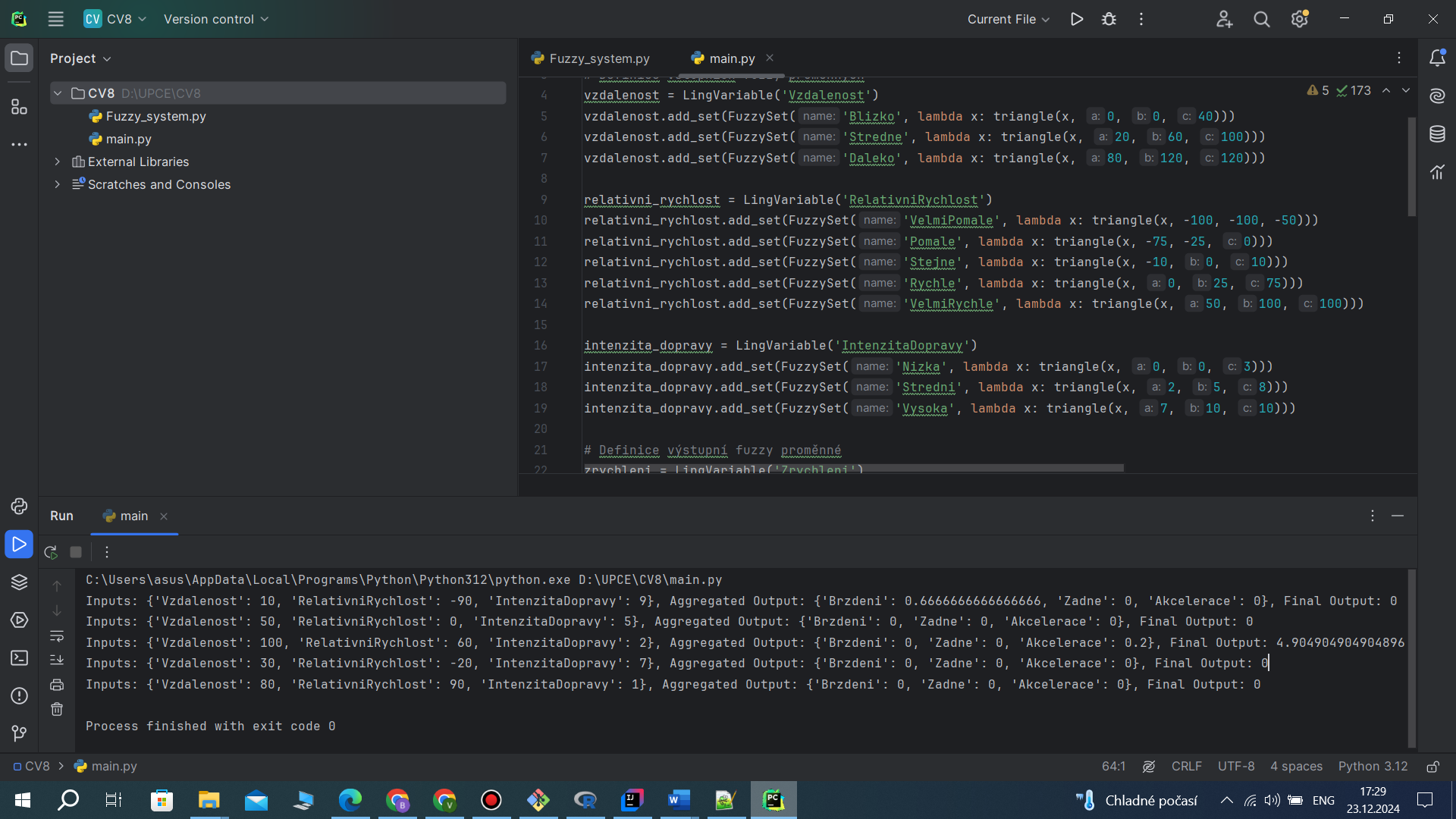
**]**

**for** inputs **in** test\_inputs**:**

aggregated\_output **=** fuzzy\_system**.**infer**(**inputs**)**

final\_output **=** fuzzy\_system**.**defuzzify**(**aggregated\_output**)**

**print(**f"Inputs: {inputs}, Aggregated Output: {aggregated\_output}, Final Output: {final\_output}"**)**



**Fuzzifikace vstupů**

Fuzzifikace je proces převodu číselných hodnot vstupních proměnných do fuzzy množin. Tento program pracuje se třemi vstupními proměnnými:

* **Vzdálenost**: Hodnota vzdálenosti od překážky (0–120 m). Množiny:
  + Blizko, Stredne, Daleko
* **Relativní rychlost**: Rychlost vůči předchozímu vozidlu (-100 až 100 km/h). Množiny:
  + VelmiPomale, Pomale, Stejne, Rychle, VelmiRychle
* **Intenzita dopravy**: Hustota dopravy (0–10). Množiny:
  + Nizka, Stredni, Vysoka

Každá množina má definovanou členovou funkci (přes funkci triangle), která vrací stupeň příslušnosti dané hodnoty k množině. Například:

vzdalenost **=** 30 # Hodnota vstupu

fuzzy\_value\_blizko **=** triangle**(**30**,** 0**,** 0**,** 40**)** # Vystup: stupen prislusnosti k mnozine 'Blizko'

**Vyhodnocení pravidel (Inference)**

Program definuje pravidla, která propojují vstupní fuzzy množiny s výstupní fuzzy množinou Zrychleni.

Pravidla používají logiku „min“ pro výpočet aktivace. Aktivace pravidla je nejnižší stupeň příslušnosti všech vstupních množin v podmínce pravidla. Například:

aktivace **=** **min(**prislusnost**(**Vzdalenost**=**Blizko**),**

prislusnost**(**RelativniRychlost**=**VelmiPomale**),**

prislusnost**(**IntenzitaDopravy**=**Vysoka**))**

### ****Agregace výstupních množin****

Pro každé pravidlo je aktivace použita k určení příslušnosti výstupní fuzzy množiny (Zrychleni) na základě pravidla. Výstupní fuzzy množiny (Brzdeni, Zadne, Akcelerace) se pak agregují přes všechna pravidla pomocí logiky „max“.

### ****Defuzzifikace****

Defuzzifikace je proces převodu fuzzy množin zpět na číselnou hodnotu zrychlení v rozsahu [-5, 5] m/s². Použitá metoda je hledání maxima (CoG – Center of Gravity). Program generuje body z intervalů fuzzy množin, počítá hodnoty členství a vrací bod s nejvyšším váženým členstvím.

**for** xi **in** x**:** # Pro každý bod v rozsahu [-5, 5]

**for** set\_name**,** membership\_value **in** aggregated\_output**.**items**():**

membership\_value\_temp **=** **min(**self**.**output\_variable**.**sets**[**set\_name**].**membership**(**xi**),** membership\_value**)**

**if** membership\_value\_temp **>** max\_membership**:**

max\_membership **=** membership\_value\_temp

max\_xi **=** xi

Výstupní hodnota max\_xi je výsledné zrychlení.

### ****Testovací scénáře****

Na konci program testuje různé kombinace vstupních parametrů (např. Vzdalenost=10, RelativniRychlost=-90, IntenzitaDopravy=9) a vypisuje:

* Agregovaný výstup fuzzy množin (např. {Brzdeni: 0.7, Zadne: 0.0, Akcelerace: 0.0}).
* Defuzzifikovanou hodnotu zrychlení (např. -4.2 m/s²).

Program simuluje logiku řízení autonomního vozidla tak, aby reagovalo na dopravní podmínky a nastavilo zrychlení podle:

* Vzdálenosti od překážky.
* Relativní rychlosti vůči překážce.
* Hustoty dopravy.

Pro spuštění programu je nutné spustit main.py