ExamplesFuzzyExpertSystems

June 27, 2023

[ ]:
```
from experta import *
import numpy as np
from IPython import display
import UPAFuzzySystems as UPAfs
import matplotlib.pyplot as plt
```

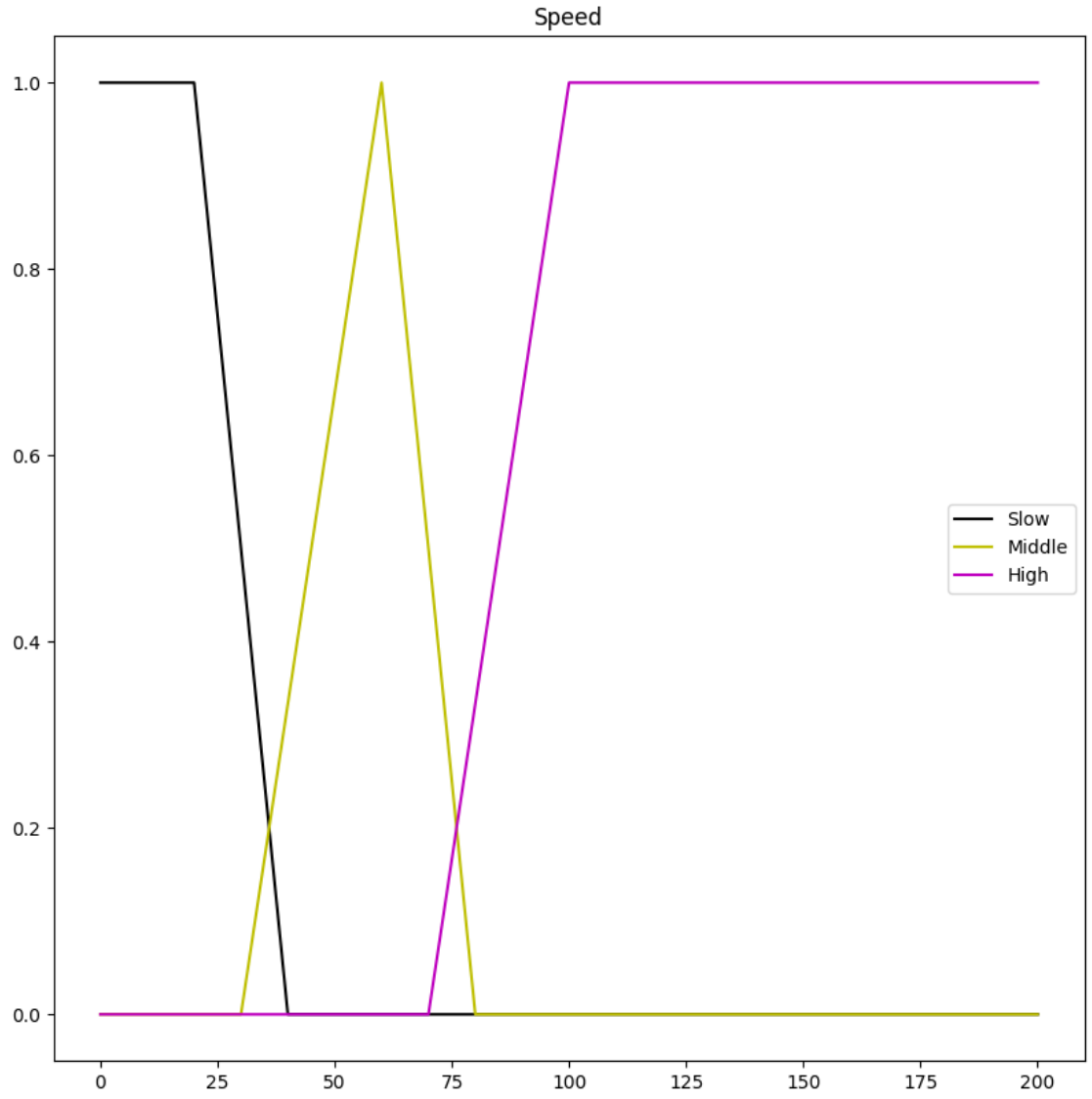
[ ]:
```
class Car_Speed(Fact):
    pass
```

[ ]:
```
class Collision_distance(Fact):
    pass
```

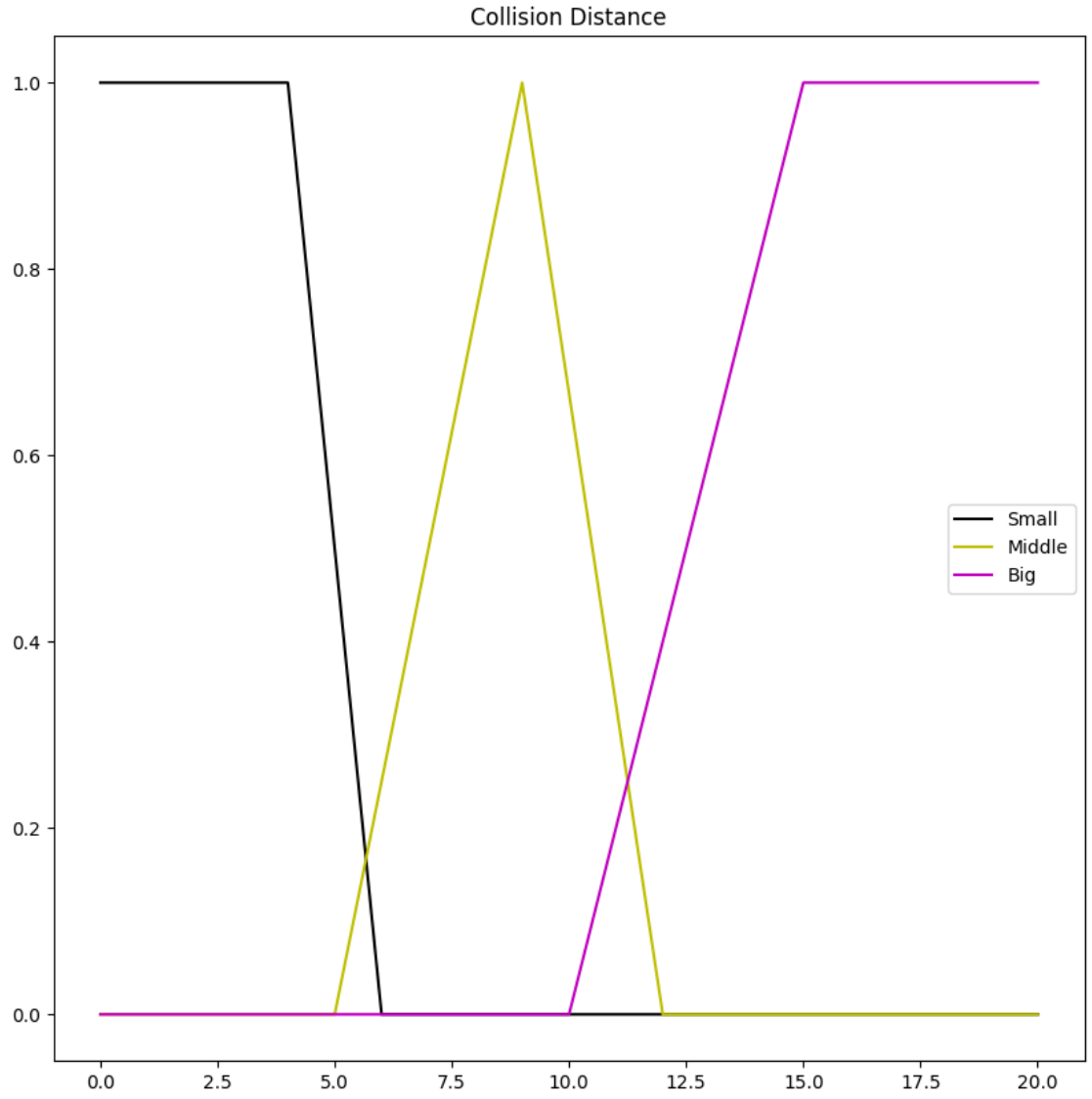
[ ]:
```
class Breakes_force(Fact):
    pass
```

[ ]:
```
class Impreso(Fact):
    pass
```

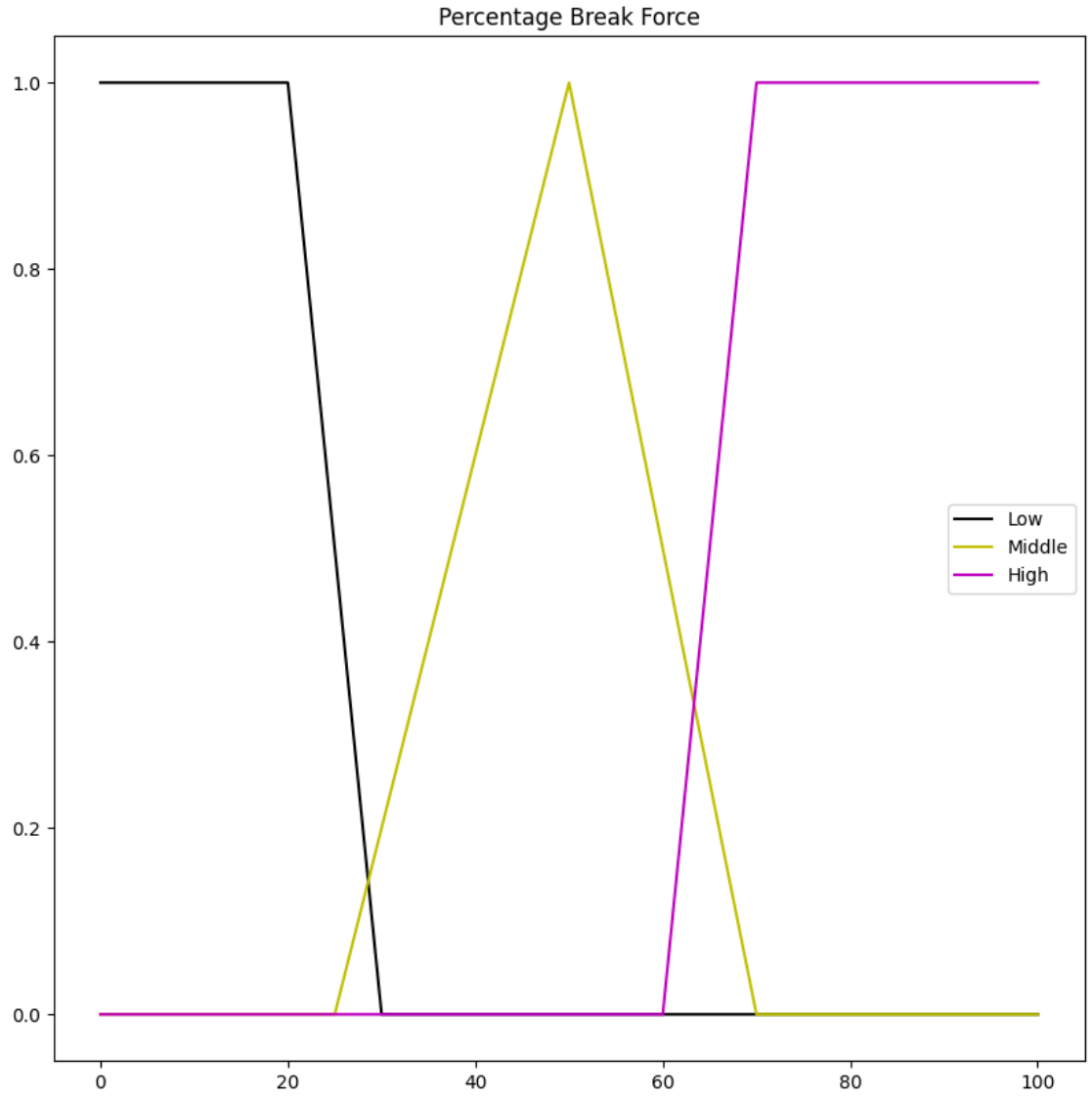
[ ]:
```
speed_universe = np.arange(0,200.1,0.1)
CarSpeedUniverse=UPAfs.fuzzy_universe('Speed',speed_universe,'continuous')
CarSpeedUniverse.add_fuzzyset('Slow','trapmf',[0,0,20,40])
CarSpeedUniverse.add_fuzzyset('Middle','trimf',[30,60,80])
CarSpeedUniverse.add_fuzzyset('High','trapmf',[70,100,200,200])
CarSpeedUniverse.view_fuzzy()
```



[ ]:
```
collision_distance_universe = np.arange(0,20.1,0.1)
CollisionDistanceUniverse=UPAfs.fuzzy_universe('Collision␣
 ↪Distance',collision_distance_universe,'continuous')
CollisionDistanceUniverse.add_fuzzyset('Small','trapmf',[0,0,4,6])
CollisionDistanceUniverse.add_fuzzyset('Middle','trimf',[5,9,12])
CollisionDistanceUniverse.add_fuzzyset('Big','trapmf',[10,15,20,20])
CollisionDistanceUniverse.view_fuzzy()
```



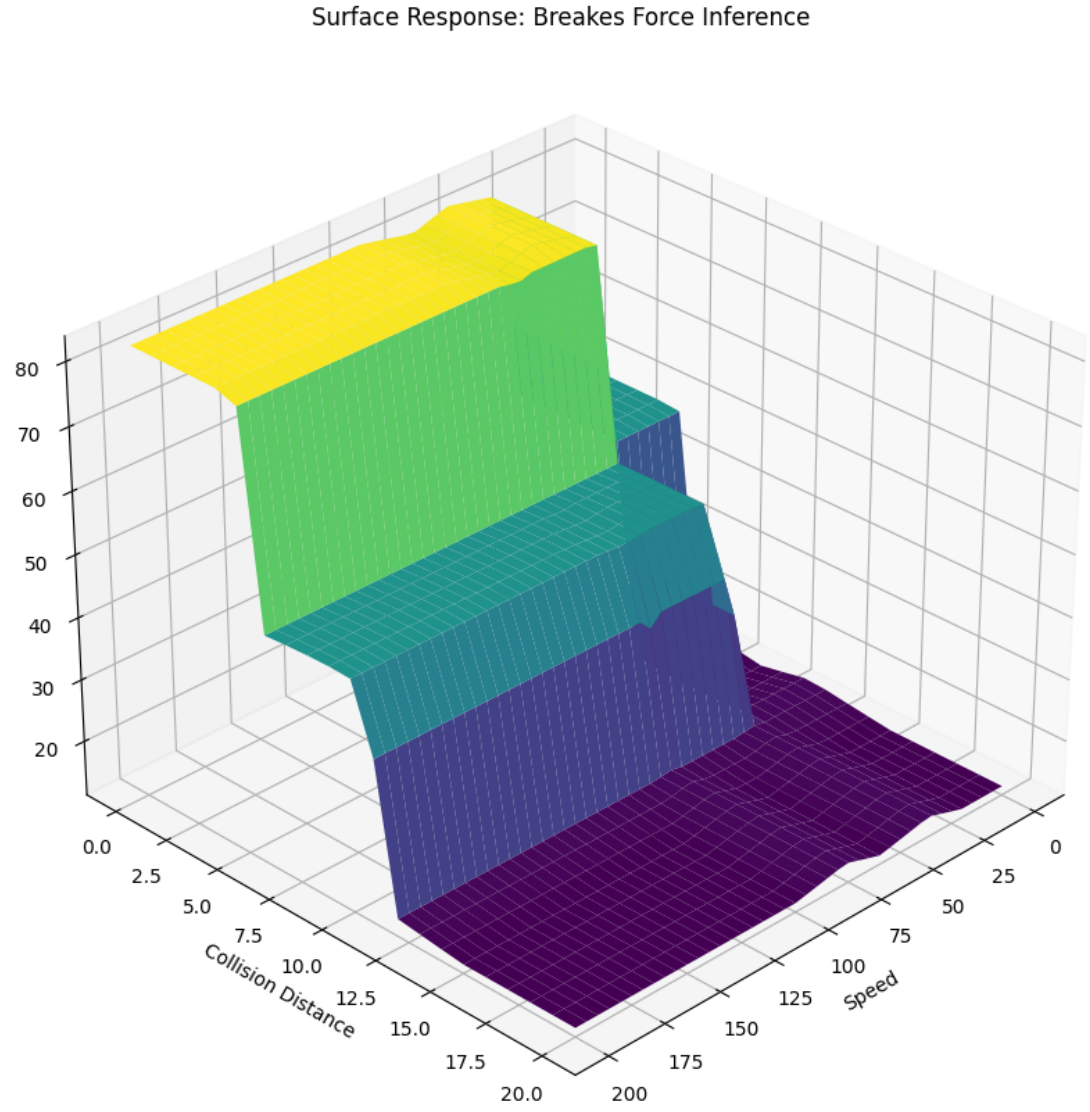
[ ]:
```
break_force_universe = np.arange(0,100.1,0.1)
BreakForceUniverse=UPAfs.fuzzy_universe('Percentage Break␣
 ↪Force',break_force_universe,'continuous')
BreakForceUniverse.add_fuzzyset('Low','trapmf',[0,0,20,30])
BreakForceUniverse.add_fuzzyset('Middle','trimf',[25,50,70])
BreakForceUniverse.add_fuzzyset('High','trapmf',[60,70,100,100])
BreakForceUniverse.view_fuzzy()
```



[ ]:
```
Breakes_Force_Inference = UPAfs.inference_system('Breakes Force Inference')
Breakes_Force_Inference.add_premise(CarSpeedUniverse)
Breakes_Force_Inference.add_premise(CollisionDistanceUniverse)
Breakes_Force_Inference.add_consequence(BreakForceUniverse)

Breakes_Force_Inference.add_rule([['Speed','Slow'],['Collision␣
 ↪Distance','Small']],['and'],[['Percentage Break Force','Middle']])
Breakes_Force_Inference.add_rule([['Speed','Slow'],['Collision␣
 ↪Distance','Middle']],['and'],[['Percentage Break Force','Low']])
Breakes_Force_Inference.add_rule([['Speed','Slow'],['Collision␣
 ↪Distance','Big']],['and'],[['Percentage Break Force','Low']])
```

```
Breakes_Force_Inference.add_rule([['Speed','Middle'],['Collision␣
 ↪Distance','Small']],['and'],[['Percentage Break Force','High']])
Breakes_Force_Inference.add_rule([['Speed','Middle'],['Collision␣
 ↪Distance','Middle']],['and'],[['Percentage Break Force','Middle']])
Breakes_Force_Inference.add_rule([['Speed','Middle'],['Collision␣
 ↪Distance','Big']],['and'],[['Percentage Break Force','Low']])

Breakes_Force_Inference.add_rule([['Speed','High'],['Collision␣
 ↪Distance','Small']],['and'],[['Percentage Break Force','High']])
Breakes_Force_Inference.add_rule([['Speed','High'],['Collision␣
 ↪Distance','Middle']],['and'],[['Percentage Break Force','Middle']])
Breakes_Force_Inference.add_rule([['Speed','High'],['Collision␣
 ↪Distance','Big']],['and'],[['Percentage Break Force','Low']])

Breakes_Force_Inference.configure('Mamdani')
Breakes_Force_Inference.build()
```

[ ]:
```
%matplotlib qt
Breakes_Force_Inference.surface_fuzzy_system([np.arange(0,205,5),np.
 ↪arange(0,21,1)])
ax = plt.gca()
ax.view_init(elev=30,azim=45)
%matplotlib inline
plt.show()
```



[ ]:
```
Breakes_Force_Inference.fuzzy_system_sim([40,15])
```

[ ]:
```
array([[14.1578163]])
```

[ ]:
```
class SpeedCollisionControl(KnowledgeEngine):
    @DefFacts()
    def inicializar_hechos(self):
        yield Car_Speed()
        yield Collision_distance()
        yield Breakes_force()
        yield Impreso()
```

```
    @Rule(NOT(Impreso(ya_impreso=W())))
    def ya_impreso(self):
        self.impresoval = Impreso(ya_impreso='no')
        self.declare(self.impresoval)

    @Rule(NOT(Car_Speed(state=W())))
    def AskCarSpeed(self):
        self.car_speed = float(input("Specify the car speed"))
        self.car_speed_dec = Car_Speed(state=self.car_speed)
        self.declare(self.car_speed_dec)

    @Rule(NOT(Collision_distance(state=W())))
    def AskCollisionDistance(self):
        self.collision_distance = float(input("Specify the car collision␣
 ↪distance"))
        self.collision_distance_dec = Collision_distance(state=self.
 ↪collision_distance)
        self.declare(self.collision_distance_dec)


    @Rule(Collision_distance(state=P(lambda x:x<=20) | P(lambda x:x>=0)),
          Car_Speed(state=P(lambda x:x<=200) | P(lambda x:x>=0)),
          Impreso(ya_impreso='no'))
    def ReturnBreakeForce(self):
        BreakesForce = Breakes_Force_Inference.fuzzy_system_sim([self.
 ↪car_speed,self.collision_distance])
        print(f"Press the break {BreakesForce[0][0]} %")
        self.modify(self.impresoval,ya_impreso='si')



engine = SpeedCollisionControl()
engine.reset()
engine.run()
```

```
Press the break 48.194435472177354 %
```