LÁGY SZÁMÍTÁSTECHNIKAI MÓDSZEREK ALKALMAZÁSÁNAK LEHETŐSÉGE A FUZZYTECH5.5 SZOFTVERREL

Simon Béláné dr. Balogh Ágnes, simona@nyf.hu

Nyíregyházi Főiskola, 4401 Nyíregyháza, Sóstói út 31.

Abstract

In the paper we summarized the possibilities of the software FuzzyTECH5.5 purchased from INFORM GmbH. We can use it for quick online design of fuzzy logic systems and neuro-fuzzy systems. We overviewed the features of the linguistic variables, rule blocks, membership functions, inference and defuzzification methods, code generation options, system optimization tools. Additionally we showed the sample system of the controlling a container crane.

1. Bevezetés

A lágy számítástechnika meghatározás Lotfi A. Zadeh-től származik. Ő indította el 1990-ben a BISC = Berkeley Initiative in Soft Computing projektet, amelynek célja világ-első kutató központ létrehozása volt a soft-computing, magyarul lágy számítástechnika területén alap és alkalmazott kutatásokban. Jelenleg a BISC csoportban közel 600-an dolgoznak: hallgatók, tanárok, magán és nem magán szervezetek alkalmazottai. A tagság feltétele érdeklődés a soft computing iránt. A soft computing(SC) – Zadeh szerint, egy szignifikáns paradigmaváltást képvisel a számítástechnika céljait illetően. Ő már 1976-ban bevezette a nyelvi változó fogalmát.A váltás azt a tényt tükrözi, hogy az emberi ész képes tárolni és feldolgozni olyan információt is, ami kevésbé precíz, kevésbé biztos és nem kategorizálható élesen. A számítógépet is fel lehet vértezni ezzel a képességgel. Ennek a képességnek az a titka, hogy ne csak számokkal, hanem szavakkal is tudjon számolni a gép. Zadeh szerint a lágy számítástechnika főbb összetevői: a fuzzy logika, a neurális hálók, és a valószínűségi következtetés (Probabilistic Reasoning) területei.

Az EU 5. Keretprogramjában megvalósított EUNITE projekt (*EUropean Network of Intelligent TEchnologies for Smart Adaptive Systems*), ami 2001-be indult, ezekre a területekre, mint intelligens módszerekre hivatkozik, amelyekkel intelligens alkalmazkodó rendszereket lehet fejleszteni. Az első európai méretű vita eredményét Dr. David Anguita DIBE-Genovai egyetem oktatója foglalta össze *Smart Adaptive Systems: State of the Art and Future Directions of Research* című dolgozatában. P.Angelov UK, P. Berka CZ, G. Dorffner AT, S. Pizzuti I, J.L. Verdegay ES, kutatók véleményeinek bevonásával.

Amint az kiderül az európai és amerikai kutatási és fejlesztési irányzatokból, a fuzzy logika, az ideghálók és egyéb biológia által inspirált matematikai, számítástechnikai módszerekkel foglalkozni ma egyet jelent az információs társadalom építőkockáinak gyártásával.

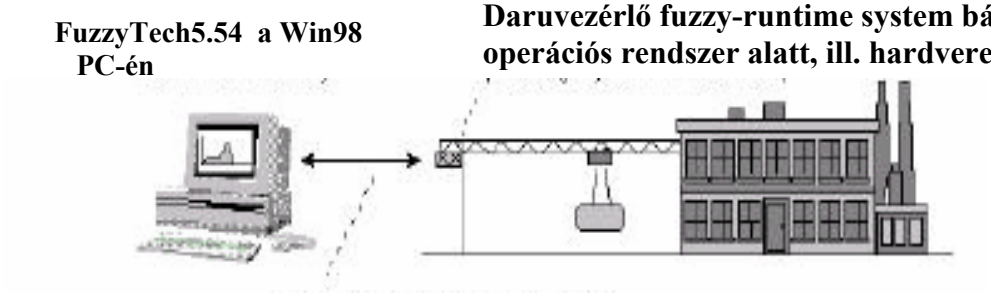
Ilyen építőkocka készítésén dolgozunk mi is. A közelmúltban OTKA műszerpályázati támogatásból megvásároltunk egy fuzzy logikai rendszert fejlesztő szoftvert. Jelenleg ennek a FuzzyTECHnevű program 5.54-es verziójának a lehetőségeit tanulmányozzuk. Eddig szerzett tapasztalatainkból adunk most közre bizonyos részeket.

2. A fuzzyTech program származása és online változata

A fuzzyTech programot az INFORM GmbH cég fejlesztette ki, amelyet 1987-ben Zimmermann professzor alapított Aachen-ben. A program 1990 óta a világ vezető programja a fuzzy logikai, a fuzzy vezérlési, és a neuro-fuzzy rendszerek tervezésére, fejlesztésére kifejlesztett programok között. Az INFORM partnerei Foxboro, AllenBradley, ABB, Bosch.

Ez a programváltozat elsősorban a technikai, műszaki vezérlések, alkalmazások fejlesztését támogatja, akár on-line módon is, kiegészítve a neuro-fuzzy modullal és klaszterező modullal. Létezik azonban pénzügyi alkalmazásokat fejlesztő változat.

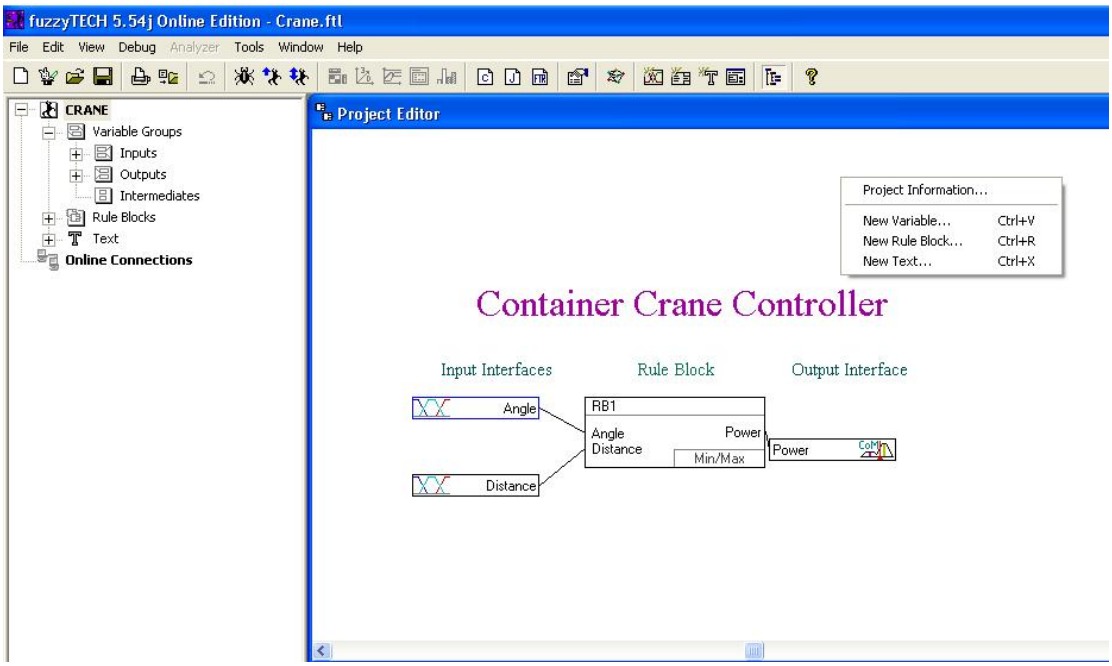
Mi egy online változatnak oktatási verzióját installáltuk egyetlen Pentium II. PC-én Win98 operációs rendszer alatt. Hardverkulccsal tudjuk használni elmentések esetén. A program fejlesztő modulja és az általa kifejlesztett runtime rendszerek két különböző PC-én is futtathatók online kapcsolatban. Többféle csatornán is létesülhet kapcsolat. Támogatja a soros RS232 interfészt, a DDE kapcsolatot, IPX/SPX csatornát és a TCP/IP protokollt, valamint saját definiálású csatorna (FTOCC) kiépítésére is ad lehetőséget. Lásd 1. ábra.



1. ábra
Online fejlesztés, a célrendszer és a fejlesztőrendszer
közötti kommunikációs csatorna segítségével

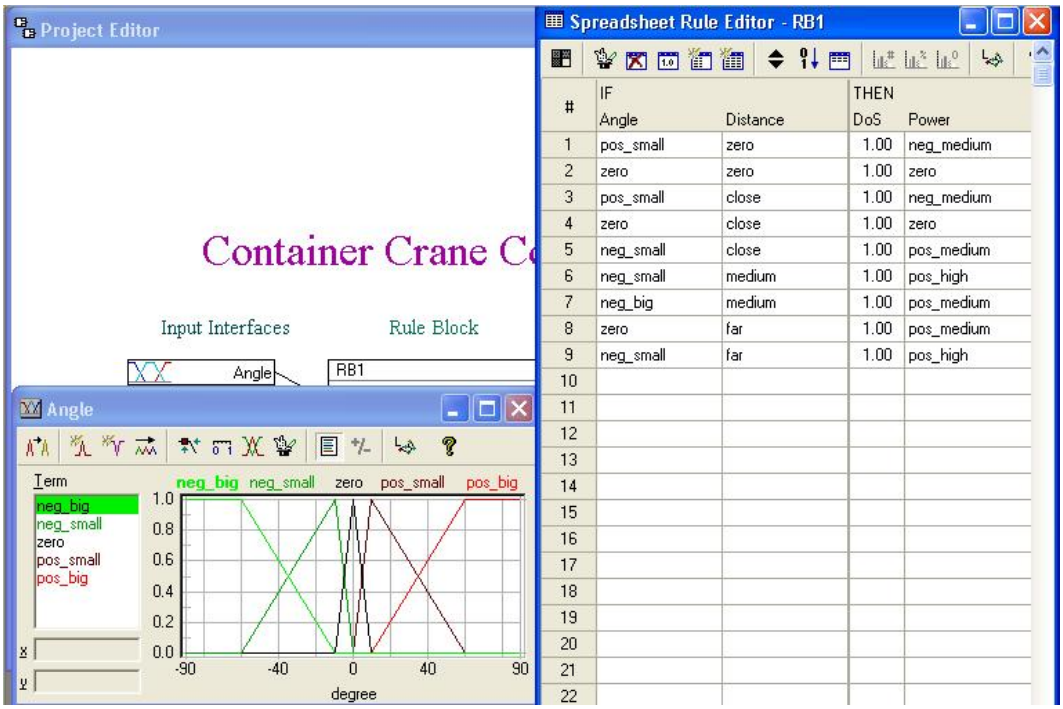
Amikor egy saját rendszert akarunk fejleszteni, első lépésként a fejlesztői shell programmal vizuálisan megtervezzük a rendszert. Vagy a tervező varázslót használjuk, vagy a tervező asztalon egyenként előállítjuk a rendszer elemeit

Mik a fő tervezési elemek? (2. ábra)



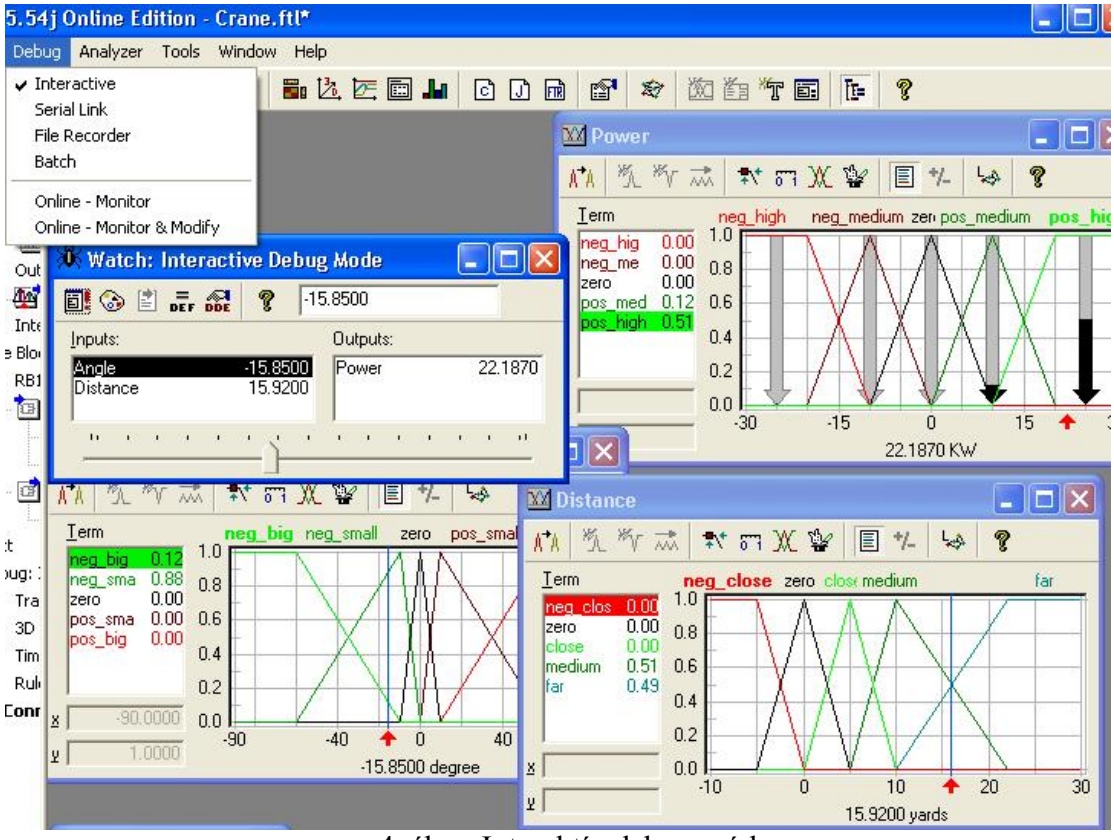
2. ábra
Tervezői eszközök

A projekt editor ablakban kettős egérkattintással a változó, a szabályblokk és szövegdoboz tervezői elemek valamelyikét adhatjuk hozzá a rendszertervünkhöz. Az egyes tervezői elemekre kattintva a saját szerkesztői ablakukban specializálhatjuk őket szándékunknak és a lehetőségeknek megfelelően. (3.ábra). A változó szerkesztőben definiálhatjuk a nyelvi változó értékkészletének minden egyes tagsági függvényét. A szabályok IF és THEN része, valamint támogatottsági fokuk is könnyedén szerkeszthető.



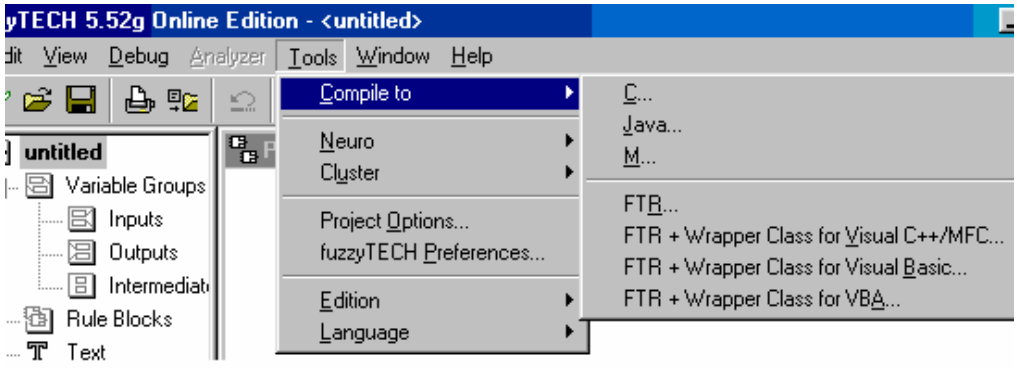
3. ábra Változó szerkesztő és szabály-blokk szerkesztő.

A File menü Save parancsával egy *.FTL (Fuzzy Technology Language) formátumú fájlban menthetjük el a fuzzy rendszer leírását. A tervezés befejezése után a tesztelés és optimalizálás lépése következik. Ezt a fejlesztési lépést is számos debug-eljárás segíti, részleteket a program sajátosságai fejezetben olvashatunk. A Debug Mode menüvel valamely eljárást elindítva minden szerkesztő dinamikussá válik, szimulálják a rendszert és kijelzik grafikusan az input adatok változtatásának hatását az outputra. A Watch ablakban numerikus mérőskála teszi könnyeddé ezt a változtatást.



4. ábra. Interaktív debug mód

A megtervezett rendszert a célplatformoktól függően különböző kódokban tudja generálni a FuzzyTECH program a Compile to parancsra (5. ábra.).



5. ábra Kódgenerálás indítása

3 A fuzzyTECH program sajátosságai

Öt táblázatban láthatjuk az online és a professzionális verzió jellemzőit. Az 1.táblázat a változókra és a szabályokra vonatkozó limiteket tartalmazza. A „Total variables" limitje azt jelenti, hogy az input, az output és a közbenső változók számának összege a teljes rendszerben nem lehet több 255-nél. Az egyes nyelvi változók értékkészlete(terms) maximum 32 fuzzy halmazzal írható le, vagy, ha nem fuzzy, akkor maximum 255 kategóriával írható le egy nyelvi változó. A szabályblokkok száma (RB) 32 lehet. Egy szabályblokkban az input változók száma és az output változók száma is maximum 11 lehet. Azaz egy szabályblokkban az input és output változók számának összege maximum 12 lehet, viszont nincs limit a szabályok összességére

Limits	Variables					Rules			
Feature / Edition	Total	Input	Output	Terms per Variable Linguistic/Categorical	Total Terms	Rule Blocks (RB)	Inputs per RB	Outputs per RB	Total Rules
Professional	255	255	32	32/255	65535	32	11 [1)]	11 [1)]	-
Online	255	255	32	32/255	65535	32	11 [1)]	11 [1)]	-

1) The total number of Inputs and Outputs per RB is limited to 12.

1. táblázat: Változók és szabályok korlátai

A második táblázatban a megengedett tagsági függvények típusairól, grafikai alakzatairól és a fazzifikálás lehetséges módszereiről tájékozódhatunk.

Standard tagsági függvény típusok: Z, Lambda, Pi, és S. Szakaszonként lineáris vagy szigmoid típusú matematikai függvényekkel lehet definiálni őket.

A „Fuzzy input" azt jelzi, hogy a változókat tagsági értékek vektoraiként visszük be. Az alapvető fazzifikációs módszer a futás közbeni számítás – a legtöbb célhardver implementációban ez a leghatékonyabb. A Look-up MBF módszer a fordítási időben végzi el a fazzifikálást és egy kereső táblaként tárolja a futtató kódban, amely sok memóriát köt le.

Kategórikus módszert akkor választunk, ha egy változó csak diszkrét egész típusú értékeket fogadhat el. (A változó tervező varázslóban a grafikájuk is más és más.)

Membership Functions	Type			Shape		Fuzzification Methods				
Feature / Edition	Standard MBF	Arbitrary MBF	Inverse MBF	linear	S-shape	Fuzzy Input	Look-up MBF [1)]	MBF Computation	Fast MBF Computation [2)]	Categorical
Professional	x	x	x	x	x	x	-	x	-	x
Online	x	x	x	x	x	x	-	x	-	x

1) *MBFs stored as look up table,* [2)] *Slope fuzzification*

2) táblázat: Tagsági függvények (MBF)

A harmadik táblázat a következtetés és a defazzifikáció módszereit foglalja össze. Egy if-then fuzzy szabályt általánosan így írhatunk le:

IF <szituáció> THEN <akció>.

Miután a fazzifikálás folyamán az input változókat átkonvertáltuk nyelvi változók értékeivé, a fuzzy következtetés lépésével tudjuk beazonosítani azokat a szabályokat, amelyek a szóban forgó szituációra érvényesülnek és ki tudjuk számítani az output nyelvi változó értékét.

Inference & Defuzzification	Aggregation Operators Standard / Compensatory				Composition		Result Aggregation		Defuzzification				
Feature / Edition	Minimum Maximum	Min-Max	Min-Avg	Gamma	Standard Rules	FAM Rules (DoS)	Max	BSUM	CoM	CoA 1)	MoM	Fuzzy Output	Hyper CoM 2)
Professional	x	x	x	x	x	x	x	x	x	x	x	x	x
Online	x	x	x	x	x	x	x	x	x	x	x	x	x

1) Fast CoA, neglects overlaps, 2) Only available as add-on module

2. táblázat: Következtetés és defazzifikáció

A *fuzzy következtetés* három számítási lépésből áll:

a.) *input aggregáció,* *b.) kompozíció,* *c.) eredmény aggregáció*.

a.) input aggregáció: A szabályok if részének <szituáció>-nak a kiszámításával megkapjuk, hogy a feltételhez viszonyítva milyen támogatottságú egy szabály, azaz a feltételek érvényesülési fokait. (kézikönyv:Fuzzy Primer 119. oldal). Ezeket az értékeket a minimum(AND aggregáció esetén) vagy maximum(OR aggregáció esetén) standard fuzzy operátorokkal számoljuk ki. Használatosak un. kompenzációs operátorok is, mint a min-max, min-avg és gamma, amelyek kompenzációs paraméterekkel (λ lambda, vagy γ gamma) alkotják meg a <szituáció> érvényesülési fokának kiszámítását.

b.) kompozíció során az egyes szabályok következmény-részének támogatottsági mértékét, az <akció> tagsági értékének kiszámításával lehet elérni. A kompozíció során a szorzat operátort használjuk: $\mu_{then} = \mu_{if} * DoS$.

Az aggregált feltétel támogatottságát megszorozzuk egy szabály fontosságát tükröző DoS értékkel. A „Standard Rules”-ban a DoS=1.0, nem változtatható. A FAM(Fuzzy Associative Map) következtetési kompozícióban a DoS érték változtatható, 0 és 1 közötti értékekre.

c) *eredmény aggregáció* során azoknak a szabályoknak az eredményét kell egyesíteni, amelyek a nyelvi változó értékkészletének ugyanazon tagjára (term-jére) tüzelnek, valamilyen mértékben. A Max operátorral kiválasztjuk a maximális tüzelési fokút, és az lesz az egyesített eredmény:

$\mu_{result} = Max_i (\mu_{then,rule\ i})$. A BSUM operátorral $\mu_{result} = \min (1, \Sigma(\mu_{then,rule\ i}))$ korlátos összeget számolunk.

A defazzifikáció során több „tüzelő” (bennmaradt, nem nulla eredményű akció) értékkészletbeli tag($term_i$) fuzzy halmazaiból kell egyetlen számértéket produkálni.

Egy standard defazzifikációs módszer CoM=Centre of Maximum: a tüzelési fokokkal súlyozott átlagát veszi a μ_{max}-okkal kiválasztható azon fuzzy halmazelemeknek, amely fuzzy halmazok az eredmény aggregáció után bennmaradtak.

Hasonlóan kompromisszumos megoldásokat alkalmaznak a többi módszerek is CoA= Centre of Area, azaz összevont terület „súlypontjának” vízszintes koordinátája lesz a konkrét számérték.

Minden fuzzyTECH program hibakereső eljárásokkal (Debug Modes) és elemző eszközökkel rendelkezik. A negyedik táblázat hibakereső, rendszer optimalizálási és rendszer analizáló eszközök összefoglalása, valamint kiegészítő modulokról ad információt.

Tools	Debug modes(internal)					Communication channels				Analyzer	Add-on modules	
Feature Edition	Inter-active	File / Batch	Serial Link	RCU, DDE	On-line	TCP/IP, IPX/SPX, DDE	Serial Interface (RS232)	SFS	User-defined (FTOCC)	TransferPlot 3D Plot, Time Plot, Trace	Neuro Fuzzy	Hyper Ininferenc e
Professional	x	x	x	x	x[4)]	x[4)]	x[4)]	x[4)]	x	x	x	x
Online	x	x	x	x	x	x	x	x	x	x	x[2)]	x

1) *Add-Ons, not included in the respective Edition,* [2)] *NeuroFuzzy Module included,* [3)] *Limited RTRCD / Trace Function,* [4)] *Only with FTRUN.DLL or fuzzyTECH Runtime Control*

4. táblázat: Tesztelési, optimalizálási eszközök

Egy kifejlesztett fuzzy rendszer tesztelése történhet off-line és on-line üzemmmódban:

Az off-line debug-golás történhet interaktív módon (lásd: 4. ábra), soros interfészen át, fájlból, RCU vagy DDE interfészeken keresztül.

A File Recorder debug módban előre rögzített adatfájllal teszteljük rendszerünket, és rekordonként elemezhetjük vizuálisan. A Batch-mód annyiban más, hogy az outputok is fájlba rögzítődnek, és további feldolgozásra vihetők.

A Monitorozó debug módban csak vizuális figyelés közben a valós adatok rögzülnek, míg az Online monitorozó és Módosító debug módban a futó rendszerben változtathatók a változók tulajdonságai, a DoS értékek, valamint letölthetjük a módosított rendszerünket a célplatformra és fordítva a célplatformról feltölthetjük az FTL kódunkat a fejlesztő PC-re.

Gyors processzek esetén lehetetlen a „röptében” módosítás, ilyenkor a Trace analizátort érdemes használni.

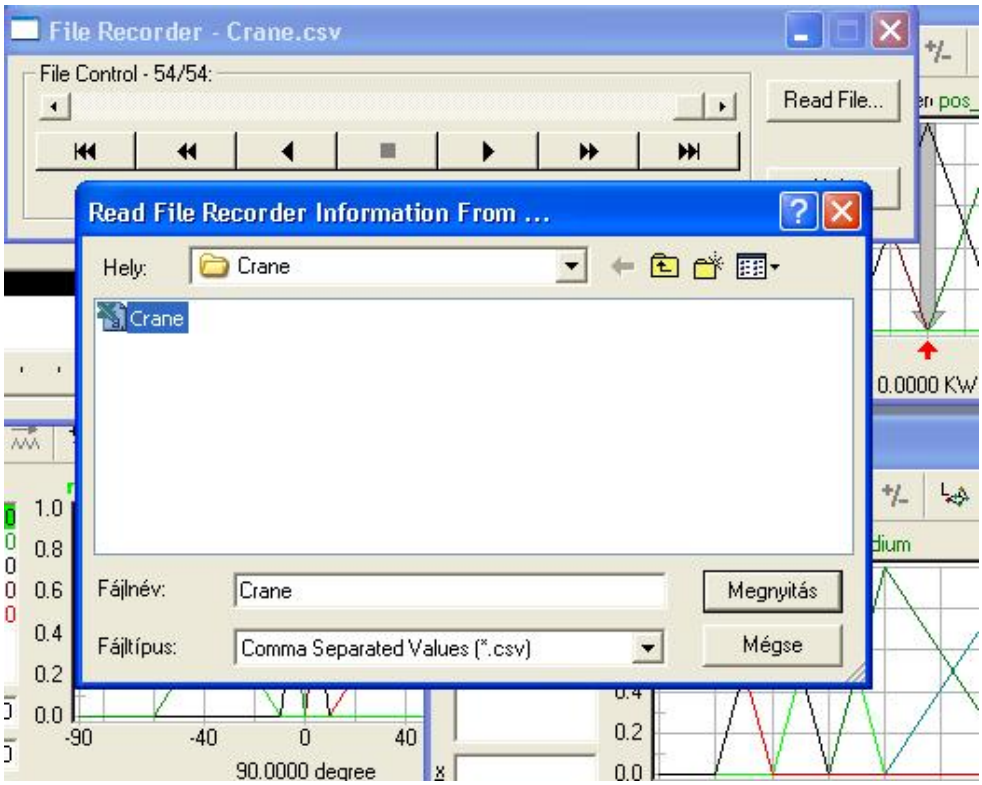
Minden debug módban használhatunk analizáló eszközöket (Analyser menüpont):

A Transfer Plot (Grafikus haladási vázlat) vizuálisan mutatja a rendszer input/output jellemzőit, egyszerre két input és egy output változó kiválasztásával síkban színmélységekkel érzékelteti az output értéket a két input kereszteződési pontjában.

A 3D Plot térben mutatja a rendszer változását egyszerre két input és egy output változó kiválasztásával, azok megengedett határain belül.

A Time Plot elemzővel a rendszer változóinak időbeni változásait követhetjük nyomon.

A Trace analizálóval egy bizonyos ideig (maximum 65535 ciklus idejéig) minden számítási ciklusból az input/output értékek adatpufferbe kerülnek, amelyből kiírathatjuk egy fájlba, és később analizálhatjuk a „File Recorder” debug módban.



6. ábra: „File Recorder” debug módban rögzített adat.

Tudnunk kell, hogy a fájlban az adatok CSV formátumúak: van egy fejléc sor(_Record_, Angle, Distance, Power mezőnevekkel) és alatta

a mezőtartalmak vesszővel elválasztva, a sorok pedig soremeléssel(ASCII 13 kóddal) következnek.

A negyedik táblázatban jelölve van még két kiegészítő modul használati lehetősége is: a neurofuzzy és a hyperinference.

A neurofuzzy modul lehetővé teszi, hogy olyan fuzzy vezérlő rendszereket fejlesszünk, amelyik tanulási képességgel rendelkeznek.

A Hyper CoM kiegészítő modul lehetővé teszi a hyperinference defazzifikációs módszer elérhetőségét, vagyis váratlan szituációkra (negatív tapasztalatra) való felkészítése a rendszernek tiltó szabályok alkalmazásával.

4. Fuzzy runtime rendszerek implementálása.

Két szempontból lehet megközelíteni az implementálást:

– Egy olyan önálló fuzzy runtime rendszert szeretnénk létrehozni, amelyet beépíthetek a saját felhasználói programomba.

– A FuzzyTECH Shell programját akarom használni arra, hogy az alkalmazásom számára kiszámítsa a fuzzy logikai rendszert.

Az első esetben a FuzzyTECH kódgenerátorát kell használni. A célplatformtól függően, – ahogy az ötödik táblázatban látjuk – különböző kódokban kaphatjuk a megtervezett fuzzy rendszer beépíthető változatait. A fuzzyTECH-nek az MCU verziói tudnak csak a kiválasztott mikrokontrollerhez optimalizált assembly és C kódot generálni.

Code Generation	Code-Options		Hardware specific Code		C-Code		Java-Code	ActiveX Java FTRUN	COBOL
Feature / Edition	Funct. Call[1)]	Public I/O	C	Assembly	ANSI	K & R	Java	FTR	COBOL [2)]
			Code Interface Resolution (8 Bit, 16 Bit, double)						
Professional	x	x	-	-	8/16/d	8/16/d	16/d	16/d	16
Online	x	x	-	-	8/16/d	8/16/d	16/d	16/d	16

[1)] *I/O passing as function parameter* [2)] *Only available as add on mudule.*

5. táblázat: Kódgenerálási lehetőségek

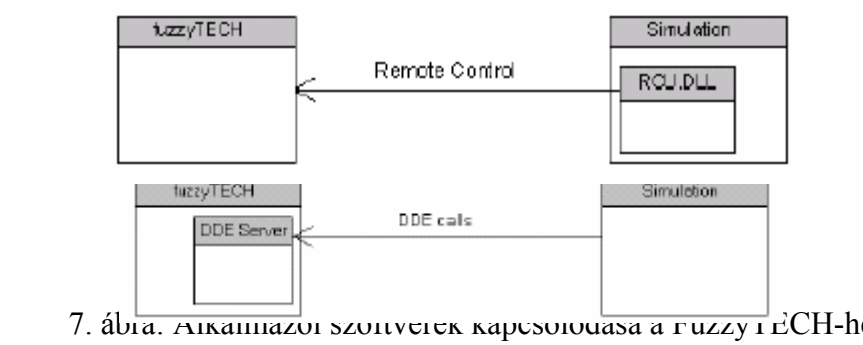
Mielőtt azonban például a C kódot generáltatnánk, a „Tools/Project Options…” menüparancsokkal felhozott dialógusablak „Code Generator” paneljén be kell állítani, hogy melyik szabvány szerint (ANSI C vagy Kernighan és Ritchie C) akarjuk generáltatni például a C nyelvű forráskódot. Ugyanitt történhet a Public I/O megjelölése is, ami azt teszi lehetővé, hogy a kódból globális változókként adhatjuk át az input és output változókat a hívó programnak. Ellenkező esetben, mint függvényparaméterek mehetnek át. Ugyancsak ebben a dialógusablakban, de a „General” panelen választhatjuk ki azt az adat típust (8 bit egész, 16 bit egész, dupla pontosságú valós), amelyet a paraméterátadásnál érvényesíthetünk a saját hívó programunkban.

A generált kódok és a saját készítésű programok számára a FuzzyTECH szolgáltatja a futtatáshoz szükséges RUNTIME könyvtárakat, amelyek tartalmazzák a szükséges alapvető fuzzy algoritmusokat forrásnyelven. Például a C kód esetén a saját C-fordítónkkal le kell majd fordítani , ettől lesz portábilis.

A generálható kódok között látjuk az .ftr típusúakat. Ezeket a modulokat a fuzzyTECH runtime DLL-je által (FTRUN32.DLL) be lehet építeni olyan MS Windows operációs rendszer alatt futó alkalmazásokba, amelyek támogatják a DLL-ek integrációját.

Az alábbi interfészek a FuzzyTECH belső számoló kernelét használják a fuzzy projektek eredményeinek a kiszámítására:

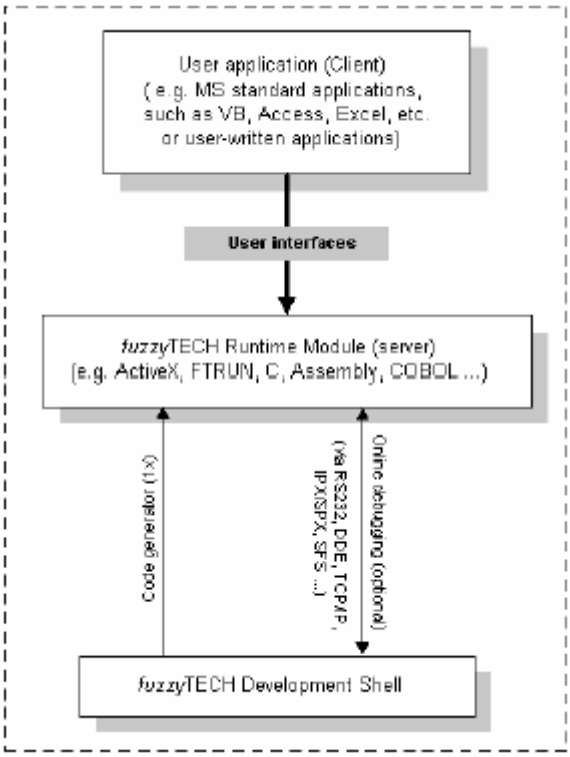
A Remote Control Unit (RCU) teszi lehetővé egy alkalmazás számára, hogy a fuzzyTECH adatokat szolgáltasson neki és elvégezze a fuzzy számításokat**.**



7. ábra: Alkalmazói szoftverek kapcsolódása a FuzzyTECH-hez

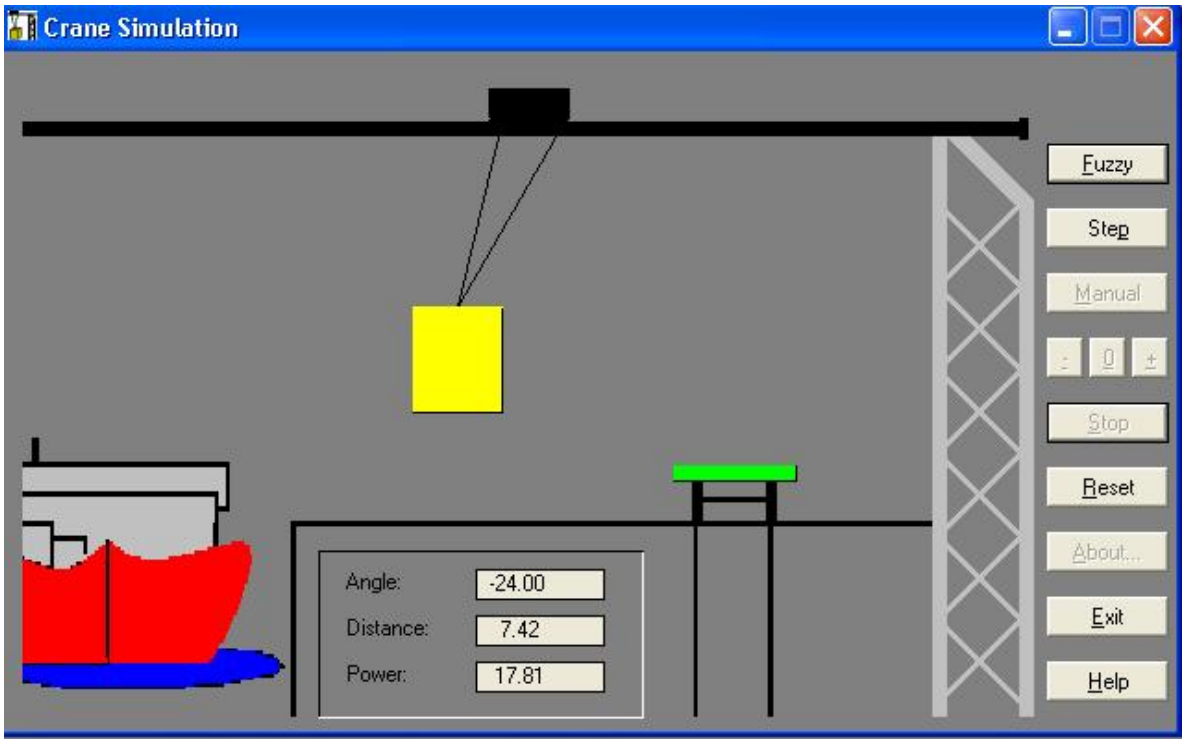
A DDE egy másik összekapcsolási mód, amely az MS Windows alkalmazásokkal köti össze a FuzzyTECH-et. A DDE kapcsolattal adatok juthatnak az alkalmazásból a FuzzyTECh-hez, és fordítva (Pl Excel –ből közvetlenül.).

Az FTRUN *fuzzy*TECH Runtime Module, egy olyan interfész program, amely a fuzzyTECH runtime kerneljét integrálja más programokkal. Az FTRUN-t mint egy 32-bites DLL-et találhatjuk meg (FTRUN32.DLL).



8. ábra: A FuzzyTECH önálló Runtime Modulja, mint szerver a fuzzy számításokhoz

5. Rakodó daru szimulációja



9. ábra: Fuzzy vezérléssel irányított rakodás szimulációja

A konténer (sárga) már felemelkedett a hajóról (vörös) és a targonca (zöld) fölé kell pozícionálni. A rakodási folyamat értékei: Szög(Angle), Távolság(Distance) és Elektromos energi(Power). A szöget és a távolságot a rakodási folyamatot szimuláló rész számolja, az energiát pedig, ami a vezérlő változó, kézzel állítjuk be (Manual), vagy a Fuzzy logikai vezérlő gombbal.

Kézi vezérlés:

A -; 0; + gombokkal lehetséges. Ha egyszer kattintunk a + gombra 0,75 kW, ha kétszer, akkor 1,5 kW-os lesz a motor teljesítménye. A lökés kilendíti, enyhén oszcilláltatja a szöget, de nem elegendő, hogy mozgásba hozza a darut a súrlódás miatt. Legalább 3kW motorteljesítmény szükséges ahhoz, hogy a daru feje elmozduljon. Most növelhetjük tovább az energiát. A 0 gomb zérus energiát ad a motornak, a – gomb negatív teljesítményt alkalmaz a fékezéshez.

Vezérlési stratégia:

Amikor nagy teljesítménnyel indítjuk a darut, a konténer nagy súlya miatt a „konténer vezeti a darut” és nem fordítva. Ez az erős visszahatása a rakománynak tipikus a konténer daruknál. Ez okozza a legnagyobb gondot az ilyen folyamatok vezérlésénél.

Darukezelés:

A darukezelőnek úgy kell pozícionálni a konténert a targonca fölé, hogy le lehessen bocsátani kilengés nélkül. Kétféle stratégia létezik erre:

- Egyik, hogy olyan lassan mozgatjuk a darufejet, hogy nem leng ki a konténer. Így soká ér célba. (Szélcsendnek kell lenni.)
- Másik egyszerű stratégia, hogy teljes erővel indítjuk és a targonca fölé állítjuk, majd kivárjuk míg lecsendesedik és akkor engedjük le. (Időjárásfüggő).

Ezek a triviális stratégiák rakodási időt rablók, s a kikötői óradíj több ezer dollár. Mivel a berakodás és a kirakodás idejét minimalizálni kell, egy olyan stratégiára van szükség, amely ellene hat a kilengésnek. Menetközben nem is probléma, de leengedni lengő rakományt nem szabad.

A legtöbb helyen a kezelő személy az, aki a daru sebességének folytonos változtatásával minimális időn belül kompenzálni tudja a kilengéseket.

Sok mérnök próbálta automatizálni ezt a vezérlési feladatot. A hagyományos PID (Proportional-Integral-Differential) vezérlés nem volt jó, mert a feladat nem lineáris. A lengés minimalizálás, csak akkor fontos, amikor a konténer közel van a célhoz. Mások matematikai modellezéssel kísérleteztek, ami ötödfokú differenciálegyenlethez vezetett, de a valóságban a következő okok miatt nem működött:

- A daru motorjának viselkedése messze nem olyan lineáris, mint ahogy azt a modell feltételezte.
- A daru feje súrlódással mozog és
- zavaró elemeket, mint a szélvihar nem tudtak beépíteni.

Nyelvi vezérlő stratégia:

Másik oldalról viszont egy kezelő ember is képes vezérelni differenciálegyenlet nélkül, pedig még egy drótkötél hossz-érzékelővel sem rendelkezik, amit egy modellalapú megoldás használna. Ő, miután megragadta a konténert, közepes motorteljesítménnyel indítja a darut, hogy lássa milyen a kilengése a konténernek. Attól függően, hogy milyen a kilengés olyan teljesítményt ad a motornak, hogy a konténer egy kicsivel a darufej mögött legyen, s maximális sebességet tud elérni minimális kilengéssel. A célhoz közeledve, a kezelő ember redukálja a motor teljesítményét, sőt negatívba viszi, hogy lefékezze. Ahogy a daru nagyon közel kerül és a teljesítményt tovább csökkentette vagy fordítva, a konténer kezd egy kicsivel a darufej elé kerülni egészen addig, amíg a konténer majdnem elérte a célt. Végül a motor teljesítményt csak annyira növeli, hogy a darufej a cél fölött legyen és a kilengés nulla legyen.

Néhány tapasztalat a vezérlő stratégiához:

1. Indítsuk közepes teljesítménnyel.
2. Ha elindult és még messze van a céltól, igazítsuk a motor teljesítményét úgy, hogy a konténer egy kicsivel a darufej mögött legyen.
3. Ha közel van a célhoz, csökkentsük a sebességet annyira, hogy a konténer kerüljön a darufej elé kicsivel.
4. Amikor a konténer a cél fölé kerül, és a kilengés zérus, állítsuk le a motort.

A nyelvi vezérlésű stratégia kivitelezése automatizálási célból:

Szenzorokat alkalmazunk a daru feje pozíciójának (Distance) és a konténer kilengés szögének (Angle) mérésére. Ezeket , mint inputokat használjuk az aktuális helyzet leírásához, és az öt tapasztalati szabályt (a 3. pontban kettő van) if-then formátumú mondatokkal leírjuk:

1. IF Distance = far AND Angle = zero THEN Power = poz_medium

2. IF Distance = far AND Angle = neg_small THEN Power = poz_high
3. IF Distance = close AND Angle = neg_small THEN Power = poz_medium
   IF Distance = medium AND Angle = neg_big THEN Power = poz_medium
4. IF Distance = zero AND Angle = zero THEN Power = zero

Általános alakja a szabályoknak: IF<szituáció>THEN <akció>

A Distance, Angle, Power három nyelvi változónak tekinthető, amelyeknek az értékkészletét meghatározhatjuk az elemeik felsorolásával, s az elemek (terms) szintén *szavak,* de nem egzakt mennyiségeket fejeznek ki, hanem egy-egy fuzzy halmazt, vagyis a „*szó*” jelentése a szóban forgó mennyiségekre nem egyformán igaz, hanem különböző mértékben. A *szó igazságértékét* kifejezhetjük 0 és 1 közötti számokkal, vagy olyan függvénnyel, ami ezeket az igazságértékeket adja. Ez a fuzzy logika, amely nem csak a két logikai értéket ismeri!

A nyelvi változó egy értékéhez egy tagsági függvény rendelhető, amelyet ebben az esetben most így jelölhetnénk: $\mu_{szó}$(szóbanforgó mennyiségek)={[0;1]}. A szóban forgó mennyiségek alatt a nyelvi változó által lefedett teljes mennyiségskálát(bázistér) kell érteni.

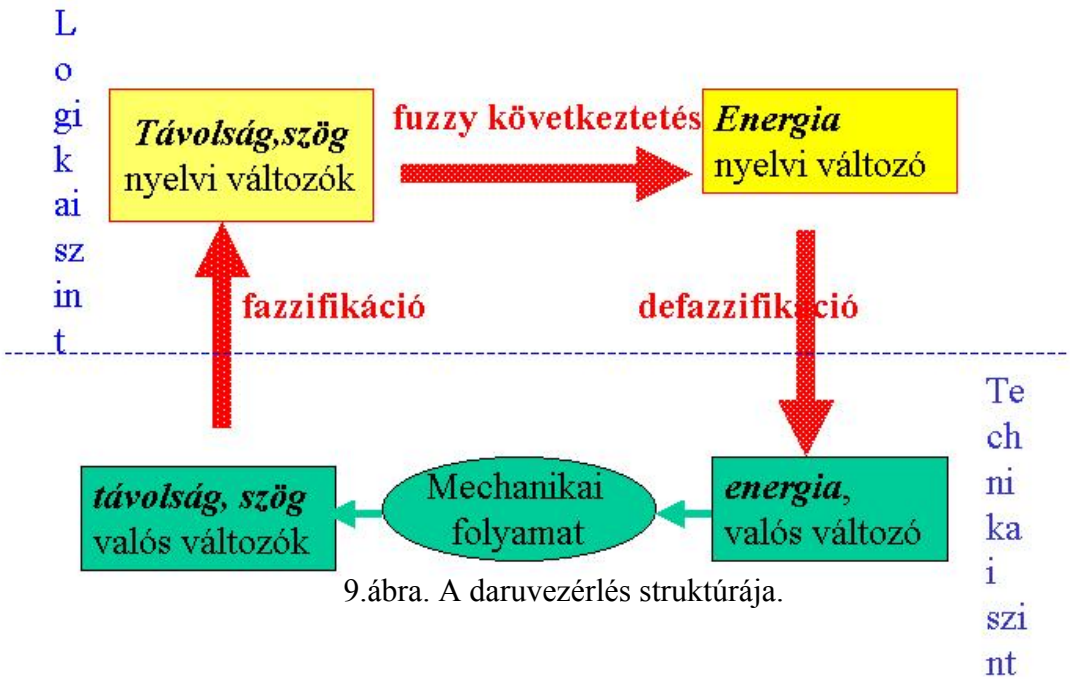
A „távolság” nyelvi változó értékkészlete például lehet a következő öt szó: negatív_közel, zérus, közel, közepes, messze.

Egy nyelvi változó definiálása tehát az értékkészlet meghatározását jelenti szavak felsorolásával, majd minden értékkészletbeli tagnak/elemnek (term) megadjuk a tagsági függvényét.

A daruvezérlés fuzzy logikai struktúrájának megértéséhez három fogalmat kell definiálnunk előbb:

1. Fazzifikáció: szenzorjelek átváltása fuzzy logikai nyelvi változó (pl távolság) értékkészletének valamely tagjává(pl közel).
2. Fazzi következtetés :
   Miután if <szituáció> then <akció> típusú szabályokból összegyűjtöttük azt a szabálykészletet, amely definiálja a rendszer viselkedését, kapunk egy un. szabálybázist.
   A következtetés, nem más, mint a szabálybázis(rule-base) kiértékelése fuzzy műveletekkel.
   Eredményeként egy nyelvi változó (pl energia) értékkészletére kapjuk a tagsági értékek megfelelő sorozatát. (Pl poz_közepes=0,4; poz_nagy=0,6; a többi nullával egyenlő). Megjegyzés: előfordul, hogy ugyanarra a *nyelvi változó értékre* két szabály is ad eredményt, ekkor aggregálni kell azokat, például a legnagyobbat választva (lásd 3. táblázat).
3. Defazzifikáció:
   A következtetés eredményeként kapott értékkészlet átváltása vezérlőjelként használható számértékre (pl 10kW).

A fenti három lépéssel leírható a logikai vezérlés folyamata. A technikai folyamattal összekötve egy körfolyamatot kapunk a daru vezérlés struktúrájára.



9.ábra. A daruvezérlés struktúrája.

A fuzzy logikának az a fő előnye, hogy egyszerű if-then relációkkal befolyásolni tudunk egy technikai rendszert. A szabálybázisban rejlő „tudásbázis" explicite kifejezhető. A mérnök, vagy egy szakember szakismerete, tapasztalata van benne nyelvi változókkal kifejezve. Ez a tudás könnyen igazolható és hatékonyan optimalizálható.

Önmaga tanítására viszont nem képes a fuzzy logikai rendszer, mint amire képes a neurális háló. A neurális háló előnyét és a fuzzy logika előnyét egyesíti a neurofuzzy módszer, amelynek alkalmazását szintén lehetővé teszi a FuzzyTECH szoftverünk.

A mintapélda visszaváltott üvegek osztályozó rendszerének kifejlesztése. A rajzos szimuláció a neurofuzzy technológia használatát demonstrálja, ahogyan az egy teljes fuzzy logikai rendszert generál pusztán a mintaadatokból. A valódi alkalmazásnak egy egyszerűsített változata van a szimulációban.

6. Összefoglalás

Összefoglalva a FuzzyTECH online verzió tanulmányozásának eddigi tapasztalatait, elmondhatjuk, hogy mind oktatási, mind kutatási és fejlesztési munkánkhoz kitűnően használható és hasznosítható programrendszer birtokába jutottunk.

Ezt az értéket szeretnénk kamatoztatni azzal is, hogy terjesztjük szűkebb hazánkban, a megyénkben is az elméleti és a tapasztalati tudásanyagunkat.

Létrehoztuk a Magyar Fuzzy Társaság Első Területi csoportját. A Csoport célja új alkalmazói rendszerek fejlesztését generálni a kis- és középvállalkozások körében.

Ezzel párhuzamosan saját programunk, a Fuzzy klaszterezés WEB-en fejlesztése is folyamatban van.

„Az IT-ipar, illetve az információ- és tudásgazdaság a világgazdaság mindinkább meghatározó részévé válik. Ahhoz, hogy legyenek sikeres, exportképes, a piaci résekbe betörni tudó vállalataink, számos lépést kell megtenni a következő években.” -idézet a MITS-ből.

7. Forráshelyek

1. www.fuzzytech.com
2. www-bisc.cs.berkeley.edu
3. www.cordis.lu/ist/results
4. www.eunite.org
5. www.sztaki.hu/sztaki/projects/html/fleet.hu.jhtml
6. pingvin.nyf.hu
7. FuzzyTECH 5.5 User’s Manual