Andrzej KOMOREK, Rafał KOWALIK, Rafał BIEŃCZAK

APLIKACJA MOBILNA WSPOMAGAJĄCA PROCEDURY LĄDOWANIA

Procedura lądowania w szczególności w bardzo trudnych warunkach należy do grypy najbardziej niebezpiecznych procedur z pośród całego loty samolotem. Mając to na uwadze autorzy podjęli próbę opracowania aplikacji softwarowej na urządzenia mobilne, która pomagać będzie pilotom w czynnościach podejścia do lądowania. Opracowane oprogramowanie oparto na sterowniku rozmytym, serwerze oraz aplikacji na urządzenia mobilne. W artykule omówiono operacie na zbiorach rozmytych, które uwzględniono w przygotowanej koncepcji oprogramowania. Ołowiowano również proces modelowania rozmytego z zawartymi regułami. Procedura podzespołów wchodzących w skład sztucznej inteligencji zasymulowano w pakiecie Matlab/ Simulink. Ponadto zaprezentowano wyniki z symulacji przeprowadzonych w tym samym oprogramowani. Całość podsumowano wnioskami nasuwającymi z wykonanych testów.

WSTĘP

Lądowanie w trudnych warunkach atmosferycznych jest najbardziej obciążającą fizycznie i psychicznie fazą lotu. Wymaga ono dobrej znajomości obowiązujących procedur, wyjątkowych umiejętności manualnych, podzielności uwagi oraz wysokiej umiejętności odbierania, przetwarzania i reagowania na bodźce zewnętrzne. Historia potwierdza, że brak wymienionych zdolności u załogi, często w połączeniu ze zmęczeniem, błędną oceną sytuacji i podjęciem nieprawidłowej decyzji mogą prowadzić do tragedii [1], [3]i.

Celem pracy jest zaprojektowanie koncepcyjnego systemu opierającego się na sterowniku rozmytym, serwerze oraz aplikacji mobilnej, który umożliwiłby wymierną i obiektywną ocenę ryzyka lądowania. Rdzeniem systemu jest sterownik rozmyty typu Mamdani zaprojektowany w pakiecie Fuzzy Logic Toolbox programu Matlab, który na podstawie wprowadzanych jako sygnały wejściowe wartości kluczowych parametrów meteorologicznych oraz wbudowanego algorytmu, wyznacza ocenę ryzyka lądowania. Aby umożliwić mobilne wykorzystanie sterownika, koniecznym jest zaprogramowanie serwera środowiska .Net, który przy pomocy bibliotek współdzielonych programu Matlab, obliczałby i wysyłał odpowiedź na zapytanie aplikacji mobilnej na platformę Android.

Niniejsza praca składa się z trzech rozdziałów. W rozdziale pierwszym przedstawiono podstawowe zagadnienia i pojęcia z zakresu najczęściej stosowanych technik sztucznej inteligencji. Rozdział drugi zawiera algorytm wraz z opisem projektu wyżej wymienionego systemu. Rozdział trzeci to podsumowanie i wnioski płynące z zaprojektowanego systemu [2].

1. SZTUCZNA INTELIGENCJA – PRZEGLĄD PODSTAWOWYCH INFORMACJI

System oparty o sztuczną inteligencję musi posiadać następujące cechy:

- Dąży do najlepszego sposobu rozwiązania problemu;
- Ma „rozumować" i działać w sposób logiczny, tzn. algorytm jego działania musi naśladować sposób wnioskowania jaki zachodzi w mózgu;
- Musi przetwarzać język naturalny na język maszyn i budować na jego podstawie bazy wiedzy;
- System inteligentny ma za zadanie samodoskonalenie i uczenie się na podstawie analogii.

Istnieje wiele definicji sztucznej inteligencji:

a) Sztuczna inteligencja jest nauką o maszynach realizujących zadania, które wymagają inteligencji, gdy są wykonywane przez człowieka (M. Minsky);
b) Sztuczna inteligencja stanowi dziedzinę informatyki dotyczącą metod i technik wnioskowania symbolicznego przez komputer oraz symbolicznej reprezentacji wiedzy stosowanej podczas takiego wnioskowania (E. Feigenbaum);
c) Sztuczna inteligencja obejmuje rozwiązywanie problemów sposobami wzorowanymi na naturalnych działaniach i procesach poznawczych człowieka za pomocą symulujących je programów komputerowych (R. J. Schalkoff) [11].

Jednym z prekursorów sztucznej inteligencji jest Alan Turing, który przeprowadził doświadczenie znane dzisiaj pod nazwą „Testu Turinga". Polegało ono na tym, że człowiek zadawał komputerowi (za pomocą klawiatury i monitora) i innemu człowiekowi te same pytania. Jeżeli uzyskane odpowiedzi były identyczny, stwierdzono by, że program jest inteligentny. Chociaż pomysł Turinga wydaje się kontrowersyjny, to test przez niego przeprowadzony faktycznie mógł w pewien wymierny sposób ocenić inteligencję maszyny. Wraz z rozwojem koncepcji sztucznej inteligencji, rozwijało się również wiele koncepcji moralnego stosunku do systemów i maszyn o nią opartych [5], [7], [8].

Termin sztuczna inteligencja został pierwszy raz zaprezentowany przez Johna McCarthy'ego w 1956r na konferencji w Dartmouth College, po której formalnie uznano sztuczną inteligencje jako dziedzinę nauki i rozpoczęto formalne badania w kierunku jej rozwoju. John McCarthy, Marvin Minsky, Allen Newell, Arthur Samuel, Herbert Simon stali się prekursorami sztucznej inteligencji. Wraz ze swoimi studentami dokonywali rzeczy przez innych ludzi podziwianych i wychwalanych. Programy i maszyny przez nich zaprojektowane potrafiły wygrać z człowiekiem w szachy, rozwiązywać problemy algebraiczne oraz dowodzić twierdzeń logicznych. Postęp w badaniach zwolnił na początku lat siedemdziesiątych, kiedy po fali krytyki niektórych środowisk matematycznych, finansowanie ze strony Departamentu Obrony Stanów Zjednoczonych bardzo zmalało. Za sprawą komercyjnego sukcesu opracowanych w latach osiemdziesiątych systemów ekspertowych, prace nad sztuczną inteligencją ponownie nabrały tempa. W późnych latach dziewięćdziesiątych i na początku XXI w. sztuczna inteligencja znalazła zastosowanie w logistyce, eksploracji danych, diagnostyce medycz-

nej i w wielu innych dziedzinach. Sukces ten zawdzięcza się coraz zwiększającej się wydajności ówczesnych komputerów oraz coraz lepszym matematycznym opisie zjawiska sztucznej inteligencji. Współcześnie, jednym z najbardziej rozpowszechnionych i najlepiej rozwiniętych technik ogólnie przypisanych do sztucznej inteligencji są sieci neuronowe, logika rozmyta oraz algorytmy ewolucyjne [9].

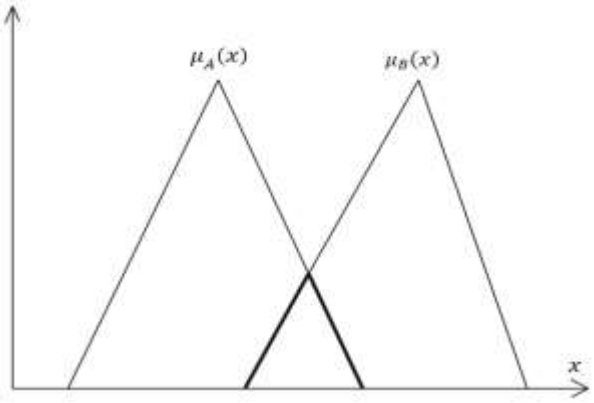
1.1. Operacje na zbiorach rozmytych

1. Przecięcie zbiorów rozmytych

Przecięciem zbiorów rozmytych $A, B \subseteq X$ jest zbiór rozmyty $A \cap B$ o funkcji przynależności

$$\mu_{A\cap B}(x) = min\left(\mu_A(x), \mu_B(x)\right), \quad (1)$$

Graficzną interpretację przecięcia zbiorów rozmytych przedstawia rysunek poniżej.

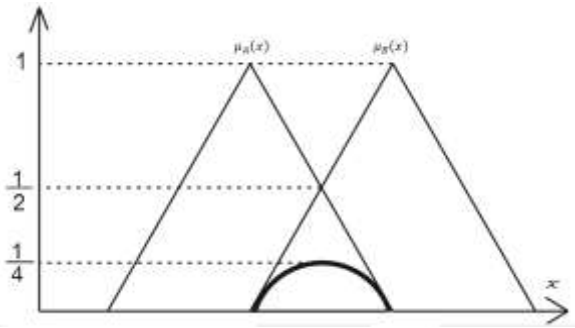


Rys. 1. *Graficzna interpretacja przecięcia zbiorów rozmytych*

2. Iloczyn algebraiczny zbiorów rozmytych

Iloczynem algebraicznym zbiorów rozmytych A i B jest zbiór rozmyty $C = A \cdot B$ zdefiniowany następująco:

$$C = \{(x, \mu_A(x) \cdot \mu_B(x)) \mid x \in X\}, \quad (2)$$

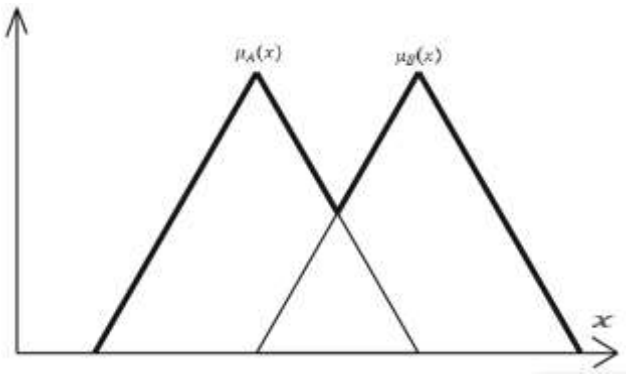


Rys. 2. *Graficzna interpretacja iloczynu zbiorów rozmytych*

3. Suma zbiorów rozmytych

Sumą zbiorów rozmytych A i B jest zbiór rozmyty $A \cup B$ określony funkcją przynależności:

$$\mu_{A\cup B}(x) = max\left(\mu_A(x), \mu_B(x)\right), \quad (3)$$

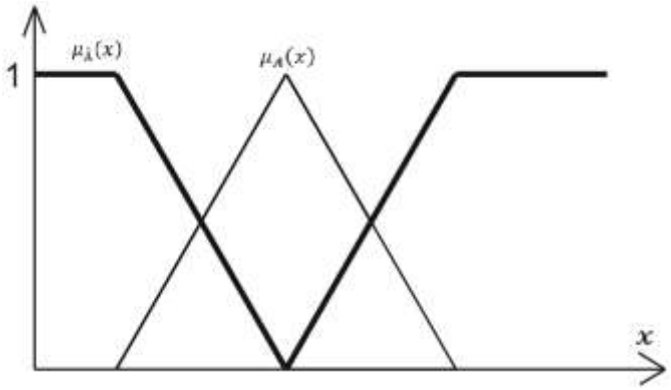


Rys. 3. *Graficzna interpretacja sumy zbiorów rozmytych*

4. Dopełnienie zbioru rozmytego

Dopełnieniem zbioru rozmytego $A \subseteq X$ jest zbiór rozmyty Â o funkcji przynależności:

$$\mu_{\hat{A}}(x) = 1 - \mu_A(x), \quad (4)$$



Rys. 4. *Graficzna interpretacja zbioru rozmytego*

1.2. Operacje na liczbach rozmytych

1) Dodawanie dwóch liczb rozmytych A_1 *i* A_2

Funkcja przynależności sumy dwóch liczb rozmytych określona jest wzorem:

$$\mu_B(y) = \sup_{y=x_1+x_2} min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2)\} \quad (5)$$

2) Odejmowanie dwóch liczb rozmytych

Funkcja przynależności różnicy dwóch liczb rozmytych określona jest wzorem:

$$\mu_B(y) = \sup_{y=x_1-x_2} min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2)\}, \quad (6)$$

3) Mnożenie dwóch liczb rozmytych

Funkcja przynależności iloczynu dwóch liczb rozmytych określona jest wzorem:

$$\mu_B(y) = \sup_{y=x_1\cdot x_2} min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2)\}, \quad (7)$$

4) Dzielenie dwóch liczb rozmytych

Funkcja przynależności ilorazu dwóch liczb rozmytych określona jest wzorem:

$$\mu_B(y) = \sup_{y=x_1:x_2} min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2)\}, \quad [11]$$

1.3. Modelowanie rozmyte

W tradycyjnym ujęciu, chcąc sterować procesami technologicznymi i różnego rodzaju maszynami, koniecznym jest określenie modelu danego procesu. Stosując do sterowania tymi procesami teorię zbiorów rozmytych, nie wymagana jest jednak znajomość ich modeli. Wystarczy jedynie sformułować zasady działania w formie **JEŻELI…TO**, które należy zaimplementować do utworzonej bazy zbiorów rozmytych. Schemat rozmytego systemu wnioskującego przedstawia rysunek poniżej:

Rys. 5. *Schemat działania układu rozmytego*

1) Blok rozmywania

Blok rozmywania (fuzyfikator) ma za zadanie zamianę konkretnej wartości sygnału wejściowego na wartość z dziedziny zbiorów rozmytych i wyznaczenia stopnia przynależności wartości wejściowej do zbioru rozmytego tych wejść.

2) Blok wnioskowania

Blok wnioskowania ma za zadanie na podstawie wypracowanych w fuzyfikatorze stopni przynależności poszczególnych sygnałów wejściowych, wypracowanie wynikowej funkcji przynależności od wszystkich sygnałów wejściowych.

Realizuje on to zadanie na podstawie bazy reguł wnioskowania utworzonej przez konstruktora oraz algorytmu wnioskowani. Baza reguł wnioskowania składa się z tzw. implikacji rozmytych, które przyjmują postać **JEŻELI…TO**, np.:

$$\boldsymbol{JEŻELI}\ x_1\ \boldsymbol{jest}\ A1\ \boldsymbol{I}\ x_2\ \boldsymbol{jest}\ B1\ \boldsymbol{TO}\ y\ \boldsymbol{jest}\ C1$$
$$\boldsymbol{JEŻELI}\ x_1\ \boldsymbol{jest}\ A1\ \boldsymbol{LUB}\ x_2\ \boldsymbol{jest}\ B1\ \boldsymbol{TO}\ y\ \boldsymbol{jest}\ C1$$

gdzie x_1 oraz y są wartościami lingwistycznymi, np. x_1 – podstawa chmur, x_2 – widzialność, y – ryzyko; A1, B1, C1 są funkcjami przynależności, np. A1 – niska, C1 – niska, C1 - wysokie.

Jeżeli przesłanki składają się z wyrażeń połączonych przez „I", to produkty logiczne są minimalną wartością odpowiednich wartości aktywacyjnych funkcji przynależności. Natomiast, jeżeli przesłanki składają się z wyrażeń połączonych przez „LUB", to konkluzja jest maksimum funkcji aktywacyjnych [3].

Wynikową funkcję przynależności dla danego sygnału wejściowego, wyliczoną na podstawie określonych stopni przynależności, najczęściej określa się używając operatora MIN, MAX, PROD, natomiast wynikowy stopień przynależności przy n-regułach formułuje się za pomocą bloku agregacji, używając najczęściej operatora MAX.

Wzory przy pomocy których określa się wynikowy stopień przynależności:

a) Interpretacja na podstawie wspólnej części zbiorów (MIN):

$$\mu_{A\to B}(x,y) = min\left(\mu_A(x), \mu_B(y)\right), \tag{8}$$

b) Interpretacja iloczynowa (PROD):

$$\mu_{A\to B}(x,y) = \mu_A(x)\,\mu_B(y), \tag{9}$$

c) Interpretacja arytmetyczna:

$$\mu_{A\to B}(x,y) = min(1, 1 - \mu_A(x) + \mu_B(y)), \tag{10}$$

d) Interpretacja minimaksowa:

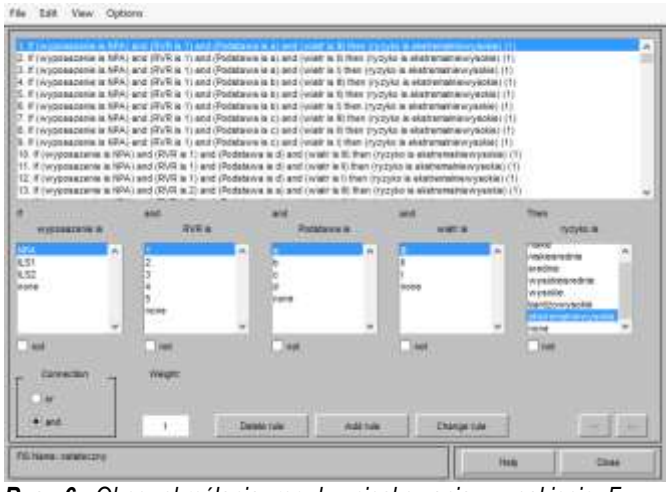
$$\mu_{A\to B}(x,y) = max\{min\left(\mu_A(x), \mu_B(y)\right), 1 - \mu_A(x)\}, \tag{11}$$

e) Interpretacja boolowska:

$$\mu_{A\to B}(x,y) = max(1 - \mu_A(x), \mu_B(y)), \tag{12}$$

Reguły wnioskowania buduje się na podstawie kilku źródeł:

- Źródła empiryczne (określenie reguł wnioskowania na podstawie doświadczenia i wiedzy konstruktora);
- Modelowanie jakościowe;
- Algorytmy automatycznego pozyskiwania wiedzy.



Rys. 6. *Okno określania reguł wnioskowania w pakiecie Fuzzy Logic Toolbox programu Matlab*

3) Blok wyostrzania

Zadaniem bloku wyostrzania (defuzyfikacji) jest określenie na podstawie wyznaczonych wcześniej zbiorów rozmytych i wyjściowych funkcji przynależności, jednej wartości należącej do zbioru liczb rzeczywistych.

Wyostrzanie może być realizowane przy pomocy następujących metod:

- Metoda center average deffuzyfication;
- Metoda center od sums deffuzifiation;
- Metoda środka ciężkości [9].

Każda metoda defuzyfikacji ma wady i zalety, a jej zastosowanie zależy od charakterystyk i wymagań stawianym projektowanemu układowi. W zależności od zastosowanej metody defuzyfikacji, wyniki końcowe mogą się różnić do kilku procent.

1.4. Próbkowanie, analiza powierzchni sterowania

Próbkowanie przeprowadza się w celu wstępnego zbadania poprawności działania projektowanego układu oraz zgodności jego zachowań z zakładanymi i przewidywanymi. Przeprowadzone poprawnie, pozwala na wykrycie ukrytych na tym etapie projektowania wad i niesprawności sterownika, a w efekcie umożliwia wyeliminowanie usterek i usprawnienie układu w procesie optymalizacji.

Dla projektowanego układu rozmytego, zastosowano dwa etapy próbkowania przedstawione w dwóch tabelach. Pierwszy etap obejmował wykonanie dziesięciu próbek dla każdej wartości sygnału wejściowego *„wyposażenie"* (w praktyce było to zbadanie odpowiedzi układu na wartości warunków meteorologicznych dla poszczególnych grup wyposażenia nawigacyjnego). Wartości sygnałów

wejściowych dobrano tak, aby przewidywana wartość sygnału wyjściowego przybierała wartości malejące. Etap ten pozwala na ewaluacje działania układu ze względu na dwa kryteria. Pierwszym z nich jest sprawdzenie, czy wraz z pogarszającymi się warunkami meteorologicznymi ocena ryzyka lądowania (wartość sygnału wyjściowego) rośnie. Kolejnym kryterium jest porównanie ocen ryzyka lądowania dla takich samych warunków meteorologicznych, ale dla różnych systemów nawigacyjnych. Założono, że dla systemu bardziej skomplikowanego, umożliwiającego lądowanie w gorszych warunkach meteorologicznych ocena ryzyka lądowania dla takich samych wartości sygnałów wejściowych powinna być niższa, niż dla systemu o gorszych parametrach. Na podstawie Tabeli 1 i 2, zastosowano następującą gradację systemów (od najmniej skomplikowanego):

a) Podejście NPA;
b) Podejście ILS Cat. I;
c) Podejście ILS Cat II.

***Tab. 1** Wyniki pierwszego etapu próbkowania*

Lp.	NPA				ILS Cat. I				ILS Cat. II			
	RVR	Podst.	Wiatr	Ryz.	RVR	Podst.	Wiatr	Ryz.	RVR	Podst.	Wiatr	Ryz.
1	0	0	40	9,64	0	0	40	9,64	0	0	40	9,64
2	250	70	36	9,64	250	70	36	9,64	250	70	36	9,64
3	500	140	32	9,5	500	140	32	9,61	500	140	32	9,5
4	750	210	28	8,99	750	210	28	7,96	750	210	28	6,86
5	1000	280	24	8,16	1000	280	24	7,23	1000	280	24	5,41
6	1250	350	20	7,33	1250	350	20	5,73	1250	350	20	4,19
7	1500	420	16	5,42	1500	420	16	4,41	1500	420	16	3
8	1750	490	12	3,78	1750	490	12	3,06	1750	490	12	2,21
9	2000	560	8	1,31	2000	560	8	1,31	2000	560	8	0,971
10	2250	630	4	0,608	2250	630	4	0,608	2250	630	4	0,608
11	2500	700	0	0,608	2500	700	0	0,608	2500	700	0	0,608

***Tab. 2** Wyniki drugiego etapu próbkowania*

Lp.	NPA				ILS Cat. I				ILS Cat. II			
	RVR	Podstawa	Wiatr	Ryz.	RVR	Podst.	Wiatr	Ryz.	RVR	Podst.	Wiatr	Ryz.
1	0	0	0	9,64	0	0	0	9,64	0	0	0	9,64
2	0	0	20	9,64	0	0	20	9,64	0	0	20	9,64
3	0	0	40	9,64	0	0	40	9,64	0	0	40	9,64
4	0	250	0	9,64	0	200	0	9,6	0	100	0	9,64
5	0	250	20	9,64	0	200	20	9,6	0	100	20	9,64
6	0	250	40	9,64	0	200	40	9,6	0	100	40	9,64
7	0	700	0	9,64	0	700	0	9,64	0	700	0	8,5
8	0	700	20	9,64	0	700	20	9,64	0	700	20	9,64
9	0	700	40	9,64	0	700	40	9,64	0	700	40	9,64
10	750	0	0	9,61	550	0	0	9,64	300	0	0	9,64
11	750	0	20	9,61	550	0	20	9,64	300	0	20	9,64
12	750	0	40	9,61	550	0	40	9,64	300	0	40	9,64
13	750	250	0	7,21	550	200	0	8,7	300	100	0	9,64
14	750	250	20	8,5	550	200	20	9,6	300	100	20	9,64
15	750	250	40	9,61	550	200	40	9,6	300	100	40	9,64
16	750	700	0	4,55	550	700	0	6,86	300	700	0	8,5
17	750	700	20	7,21	550	700	20	8,5	300	700	20	9,64
18	750	700	40	9,61	550	700	40	9,64	300	700	40	9,64
19	2500	0	0	9,64	2500	0	0	9,64	2500	0	0	9,64
20	2500	0	20	9,64	2500	0	20	9,64	2500	0	20	9,64
21	2500	0	40	9,64	2500	0	40	9,64	2500	0	40	9,64
22	2500	250	0	6,86	2500	200	0	7,36	2500	100	0	9,64
23	2500	250	20	8,5	2500	200	20	7,36	2500	100	20	9,64
24	2500	250	40	9,64	2500	200	40	8,7	2500	100	40	9,64
25	2500	700	0	0,608	2500	700	0	0,608	2500	700	0	0,608
26	2500	700	20	3,5	2500	700	20	3,5	2500	700	20	0,608
27	2500	700	40	6,86	2500	700	40	5,14	2500	700	40	3,5

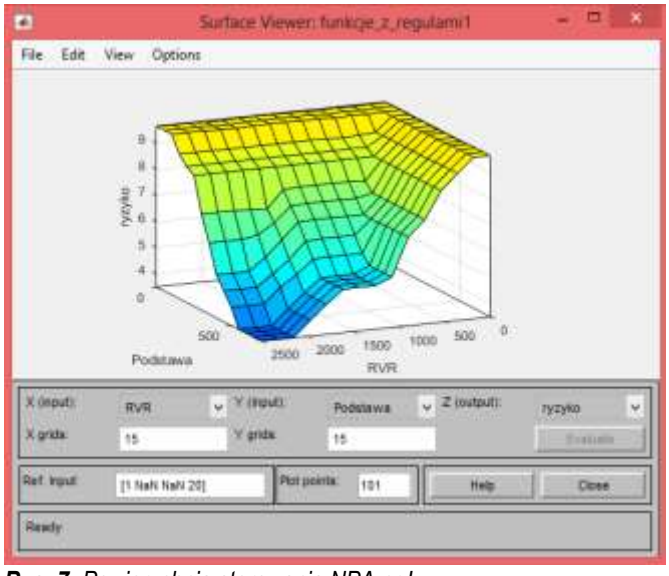
Drugi etap próbkowania przedstawiony w Tabeli 2 umożliwia sprawdzenie odpowiedzi układu dla granicznych wartości sygnałów wejściowych dla poszczególnych systemów nawigacyjnych. Graniczne wartości sygnałów wejściowych to takie wartości dla których przewidywana ocena ryzyka lądowania powinna przyjąć wartość maksymalną lub minimalną. Zastosowano następujące wartości sygnałów wejściowych i ich kombinacje:

1) Dla NPA:
   – RVR = 0, 750, 2500;
   – Podstawa = 0, 250, 700;
   – Wiatr = 0, 20, 40;
2) Dla ILS Cat. I:
   – RVR = 0, 550, 2500;
   – Podstawa = 0, 200, 700;
   – Wiatr = 0, 20, 40;
3) Dla ILS Cat. II:
   – RVR = 0, 300, 2500;
   – Podstawa = 0, 100, 700;
   – Wiatr = 0, 20, 40.

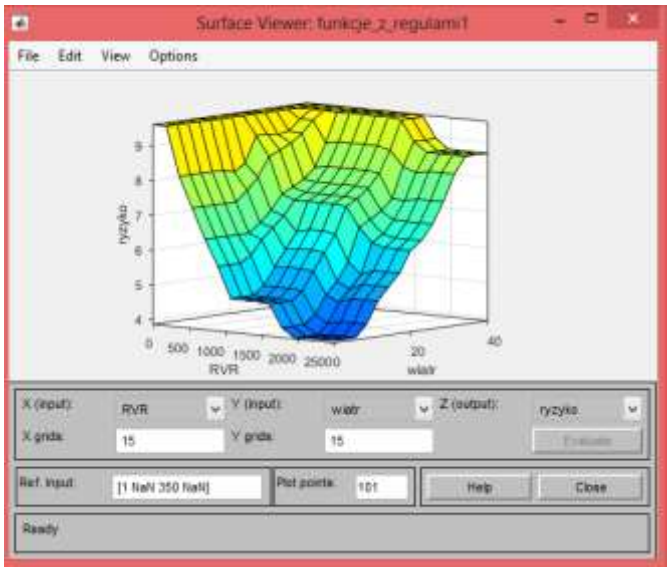
Wyniki próbkowania pokazały, że układ spełnia podstawowe założenia:

– ocena ryzyka lądowania rośnie dla pogarszających się warunków meteorologicznych niezależnie od wybranego systemu nawigacyjnego;
– porównanie ocen ryzyka lądowania dla tożsamych wartości sygnałów wejściowych potwierdziło wyżej wymienioną gradację systemów nawigacyjnych;
– ocena ryzyka lądowania przyjmuje wartości maksymalne wartości dla granicznych wartości sygnałów wejściowych.

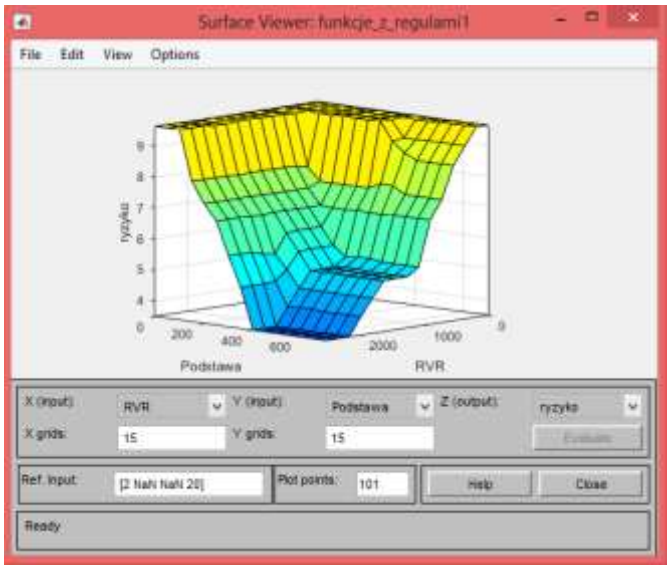
Na podstawie przeprowadzonego próbkowania wysnuto twierdzenie o poprawnym działaniu układu w ogóle. W celu potwierdzenia faktycznej poprawności działania, przeprowadzano analizę powierzchni sterowania. Pozwala ona na ogólne zbadanie działania układu i określenie trafności dobrania reguł wnioskowania [4], [6].



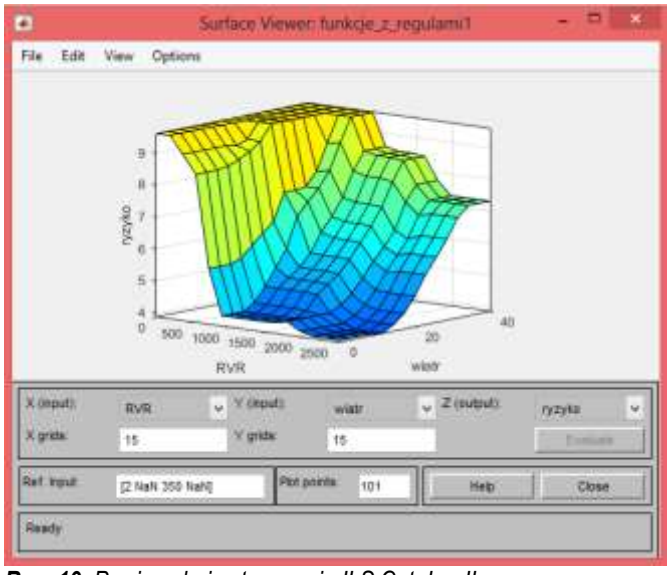
***Rys. 7.** Powierzchnia sterowania NPA nr I*



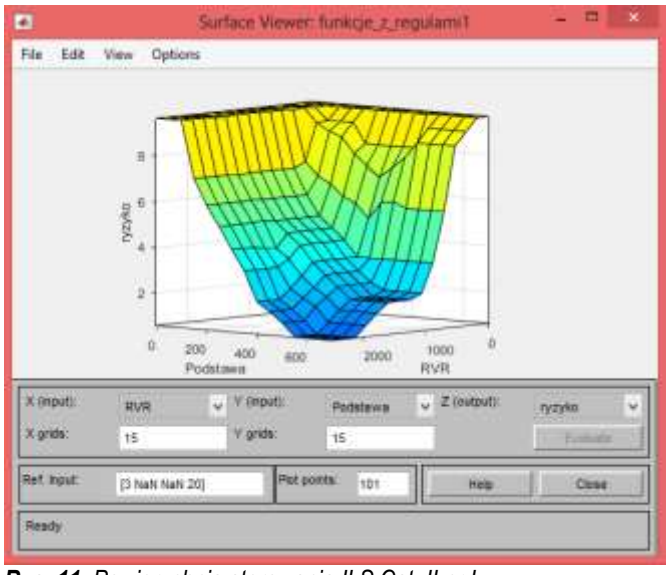
Rys. 8. *Powierzchnia sterowania NPA nr II*



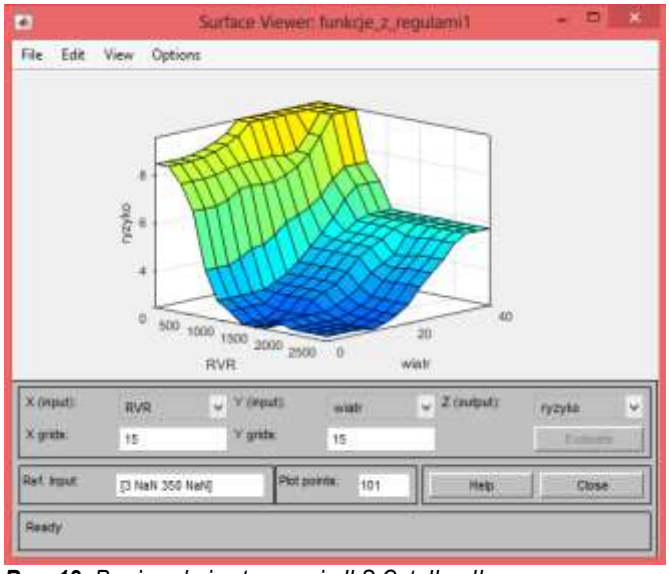
Rys. 9. *Powierzchnia sterowania ILS Cat. I nr I*



Rys. 10. *Powierzchnia sterowania ILS Cat. I nr II*



Rys. 11. *Powierzchnia sterowania ILS Cat. II nr I*



Rys. 12. *Powierzchnia sterowania ILS Cat. II nr II*

Na podstawie analizy wyżej umieszczonych powierzchni sterowania dla poszczególnych systemów nawigacyjnych, dotarto do następujących wniosków:

– niektóre powierzchnie są strome i niewygładzone;
– powierzchnie sterowania ILS Cat. II uwidaczniają niekonsekwencję oceny ryzyka – wraz ze wzrostem wartości podstawy chmur, ocena maleje, by znów zacząć rosnąć. Analogiczne zjawisko zaobserwowano przy wzroście wartości sygnału *„RVR”* (Rysunek 12).

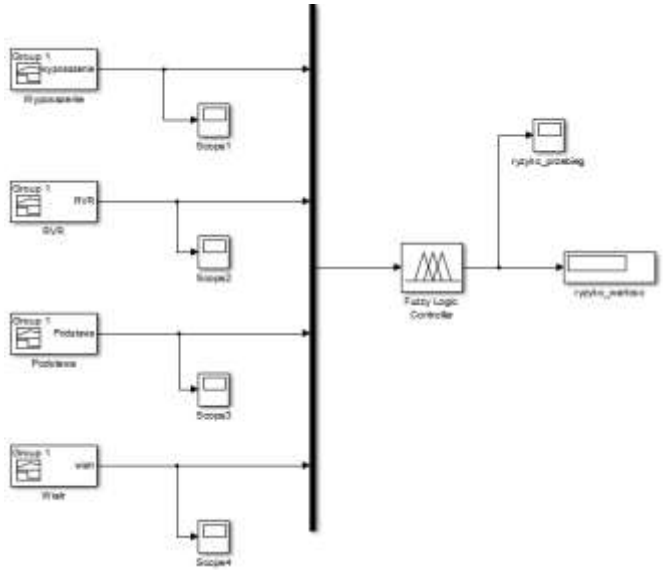
W celu wyeliminowania wyżej wymienionych błędów, poprawy jakości i poprawności działania układu, przeprowadzono optymalizację sterownika.

1.5. Symulacja w pakiecie *Simulink*

Działanie zoptymalizowanego sterownika sprawdzono w pakiecie *Simulink,* który umożliwia symulację, generowanie kodu C++ zaprojektowanego układu, testowanie i weryfikację poprawności przyjętych rozwiązań [9].

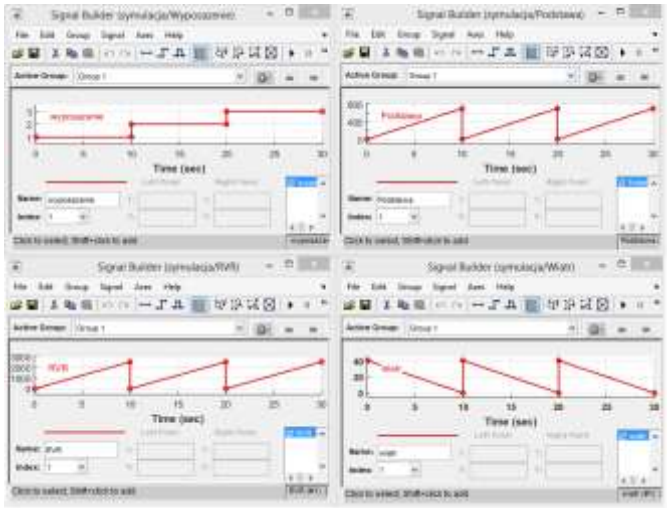
W celu przeprowadzenia symulacji, należy utworzyć schemat blokowy zaprojektowanego układu używając elementów dostępnych w załączonej bibliotece oraz zdefiniować przebiegi sygnałów wejściowych. Dla projektowanego sterownika utworzono model układu składający się z czterech modulatorów sygnałów wejściowych prze-

kazujących sygnał do kontrolera logiki rozmytej. Przebieg sygnału wyjściowego widoczny jest na oscyloskopie, a wartość na wyświetlaczu.

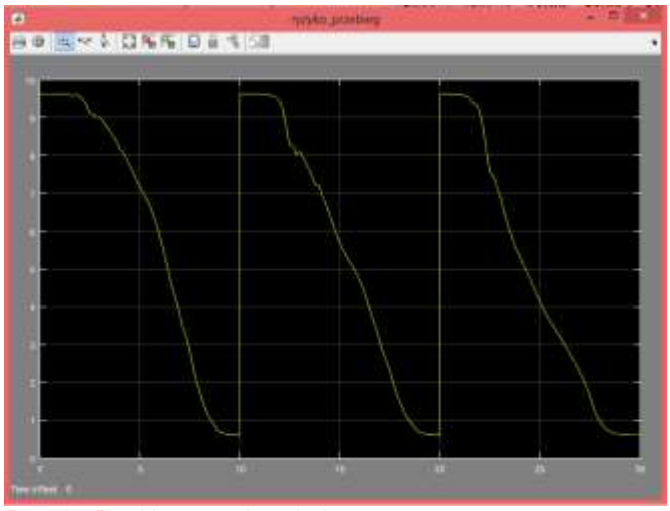


Rys. 13. *Model projektowanego układu w Simulink*

Z powodu istnienia wielu możliwości doboru przebiegów sygnałów wejściowych, istnieje tyle samo możliwości sprawdzenia poprawności działania układu. Ze względu na prostotę i łatwość interpretacji przebiegu sygnału wyjściowego, zdecydowano się na taki dobór przebiegu sygnałów wejściowych, aby przewidywana wartość sygnału wyjściowego stale malała. Symulację przeprowadzono dla czasu 30s.



Rys.14. *Przebiegi sygnałów wejściowych*



Rys. 15. *Przebieg sygnału wyjściowego*

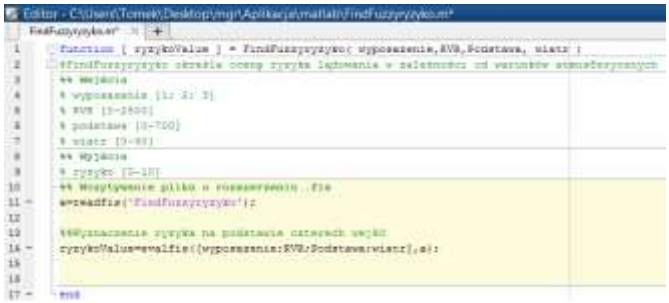
Przebieg sygnału wyjściowego ukształtował się w sposób przewidywany i potwierdził obserwacje i wnioski z przeprowadzonego próbkowania i analizy powierzchni sterowania. Dla każdego systemu nawigacyjnego ma on charakter w przybliżeniu liniowy co stanowi duża zaletę pod względem wiarygodności oceny ryzyka. Dla poszczególnych wartości sygnału wejściowego *„wyposażenie”* jest on przesunięty wzdłuż osi odciętych względem poprzedniego. Potwierdza to przyjętą gradację systemów nawigacyjnych.

Na podstawie przeprowadzonego próbkowania, analizy powierzchni sterowania oraz symulacji działania w pakiecie *Simulink*, stwierdzono, że zaprojektowany układ oceniający ryzyko lądowania w zależności od warunków atmosferycznych działa poprawnie i według zamierzeń autora. Kolejnym krokiem w utworzeniu możliwości mobilnego wykorzystania zaprojektowanego sterownika jest zaprogramowanie serwera aplikacji mobilnej. Procesy te zostały przedstawione w dalszej części artykułu.

2. PROGRAMOWANIE SERWERA

Aby korzystać z zaprojektowanego w programie *Matlab* sterownika na telefonie komórkowym wykorzystującym oprogramowanie *Android*, koniecznym było zaprogramowanie serwera wykorzystującego wyeksportowane z programu *Matlab* biblioteki współdzielone. Zaprogramowany serwer po otrzymaniu zapytania z telefonu komórkowego, przy pomocy wyeksportowanych bibliotek programu *Matlab,* oblicza odpowiedź i wysyła ją z powrotem do klienta – telefonu komórkowego [5], [8].

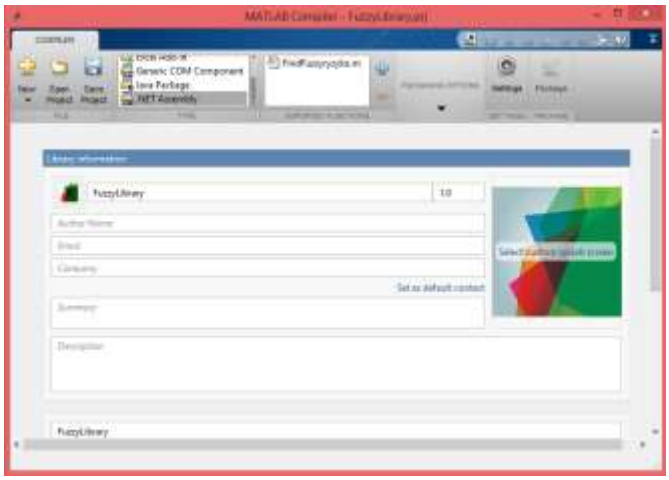
Pierwszym krokiem w zaprogramowaniu serwera było utworzenie funkcji w języku programu *Matlab* wykorzystującej zaprojektowany sterownik zapisany w pliku o rozszerzenie *.fis*. Funkcję to zapisano w pliku o rozszerzeniu *.m*, który to typ pliku wykorzystuje i obsługuje narzędzie służące do eksportowania bibliotek współdzielonych.



Rys. 16. *Funkcja wykorzystująca zaprojektowany sterownik rozmyty*

W celu wyeksportowania bibliotek współdzielonych *DLL,* wykorzystano narzędzie *Matlab Compiler*. Pozwala ono m. in. na eksportowanie bibliotek współdzielonych języka *C/C++, Java,* środowiska *.Net* oraz tworzenie autonomicznych aplikacji wykorzystujących dodatek *Matlab Runtime.* Cecha ta daje możliwość korzystania z projektów utworzonych w programie *Matlab* użytkownikom nie posiadającym programu *Matlab* i używającym innych platform.

Chcąc utworzyć bibliotekę współdzieloną, należało określić typ biblioteki *(.Net Assembly*), eksportowaną funkcję (*FindFuzzyryzyko.m*) oraz inne pliki wykorzystywane (*FindFuzzyryzyko.fis*).



Rys. 17. *Narzędzie Matlab Compiler*

Utworzone biblioteki współdzielone mogą być wykorzystane przez różnego rodzaju aplikacje. W tym przypadku wykorzystano je przy programowaniu w środowisku *.Net* serwera przyjmującego, obliczającego i odpowiadającego na zapytania aplikacji mobilnej.

.Net to platforma programistyczna utworzona przez firmę *Microsoft* umożliwiającą korzystanie z wielu języków programowania, np. *C++, C#, Visual Basic, JScript*. Składa się ona z dwóch zespołów – środowiska uruchomieniowego *Common Language Runtime (CLR)* oraz biblioteki klas *Framework Class Library (FCL). CLR* zajmuje się przetwarzaniem kodu, zarządzaniem pamięcią oraz zabezpieczeniami. Bibliotek klas to wszechstronna i szeroka kolekcja typów wielokrotnego użytku, które można wykorzystywać do tworzenia aplikacji, począwszy od tradycyjnej linii poleceń, przez aplikacje wykorzystujące graficzny interfejs użytkownika *(GUI)*, a kończąc na aplikacjach opartych o najnowsze innowacje świadczonych przez *ASP.NET*, takich jak *Web Forms* czy *XML* [7].

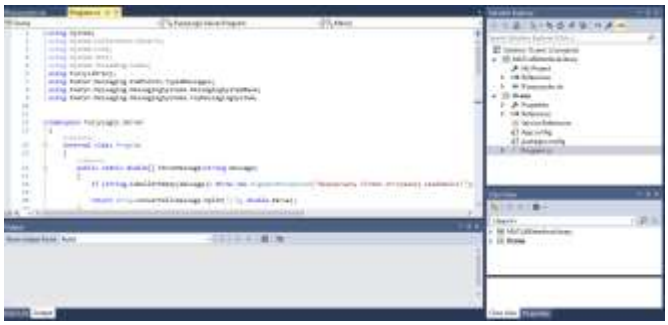
Ponieważ poszczególne zagadnienia wymagają od deweloperów używania różnych języków programowania (pewne języki dają możliwość opracowywania zagadnień matematycznych, inne są lepsze w przetwarzaniu i opisywaniu zagadnień sieciowych, czy komunikacyjnych), dużą zaletą środowiska *.Net* jest możliwość wyboru dogodnego języka programowania. Najbardziej popularnym programem służącym do operowania w środowisku *.Net* jest program *Microsoft Visual Studio*, dlatego w nim właśnie zaprogramowano serwer.

Rys. 18. *Schemat kompilacji kodu w środowisku .NET*

Projekt programowanego w *Microsoft Visual Studio* serwera, składa się z dwóch podprojektów:

- *MATLABIntefraceLibrary* napisanego w języku *Visual Basic*, który był użyty podczas tworzenia bibliotek współdzielonych *DLL* w programie Matlab oraz;
- *Ocena* napisanego w języku *C#*, w którym zdefiniowane jest działanie serwera oraz sposób łączenia z klientem.

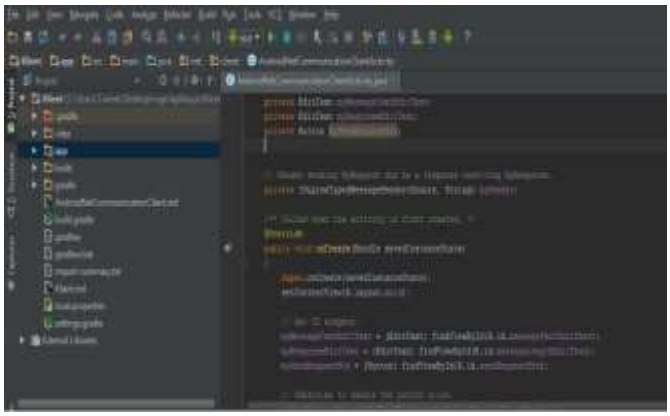
W celu umożliwienia serwerowi korzystania z algorytmów programu *Matlab*, koniecznym było dodanie odwołania do bibliotek utworzonych wcześniej w *Matlab Compiler*. Poprawne nawiązanie połączenie z klientem wymagało zainstalowania i dodania odwołania do bibliotek *Eneter Messaging Framework*. Do zaprogramowania podprojektu *Ocena* w języku *C#* użyto kodu ogólnodostępnego na stronie firmy *Eneter*.



Rys. 18. *Fragment kodu źródłowego serwera w programie Microsoft Visual Studio*

3. PROGRAMOWANIE APLIKACJI MOBILNEJ I WNIOSKI

Połączenie pomiędzy serwerem, a klientem zrealizowano przy pomocy najbardziej popularnego protokołu komunikacyjnego *TCP (Transmission Control Protocol)*. Definiuje on jak nawiązać i utrzymać niezawodne i wolne od błędów połączenie sieciowe, poprzez które aplikacje mogą przesyłać dane. *TCP* współpracuje z protokołem *IP*, który definiuje jak maszyny przesyłają pomiędzy sobą pakiety danych. Protokół *TCP* działa w trybie klient - serwer. Serwer czeka na inicjację połączenia, natomiast klient nawiązuje połączenie z serwerem. Oba protokoły *TCP/IP* są podstawą działania współczesnego Internetu.



Rys. 19*. Fragment kodu źródłowego aplikacji mobilnej*

Aplikacja mobilna na platformę Android została napisana w programie *Android Market*. W celu komunikowania się z serwerem, zainstalowano i dodano odwołania do bibliotek *Eneter Messaging Framework* oraz dodano adres *IP* zaprogramowanego serwera. Do napisania aplikacji użyto ogólnodostępnego kodu ze strony firmy *Eneter*, który zmodyfikowano do potrzeb autora.

BIBLIOGRAFIA

1. Ben Mahmoud, H., Ketata, R., Ben Romdhane, T., Ben Ahmed, S. , Hierarchical fuzzy approaches for a piloted Quality Management System", IEEE 2013 International Conference on Control, Decision and Information Technologies (CoDIT) - Hammamet, Tunisia
2. Brown M., An Introduction to Fuzzy and Neurofuzzy Systems, 1996,
3. Farrey-Goudreau, E., Wood, D.B., Fuzzy logic implementation of a RGPO jamming detector for a pilot training system, IEEE MILCOM 97 MILCOM 97 Proceedings - Monterey, CA, USA
4. Grzesik N., Podstawy sterowania rozmytego, Wyższa Szkoła Oficerska Sił Powietrznych, Dęblin; 2012,
5. Jabrane, Y., Gil Jimenez, V. P., Garcia Armada, A., Ait Es Said, B., Ait Ouhman, A. Evaluation of the effects of pilots on the envelope fluctuations reduction based on neural fuzzy systems

IEEE 2010 IEEE 11th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2010) - Marrakech, Morocco

6. Li H., Wu L., Lam H.-K., Gao, Y., Analysis and Synthesis for Interval Type-2 Fuzzy-Model-Based Systems, Springer, 2016
7. Luis M., Genetic Fuzzy Systems, World Scientific Pub Co Inc, 2001
8. Fazle Azeem M. Fuzzy Inference System: Theory and Applications, InTeO, 2012
9. Żurada J., Barski M., Jędruch W., 1996, Sztuczne sieci neuronowe, Wydawnictwo PWN, Warszawa;

USE

The procedure for landing especially in very difficult conditions is one of the most dangerous flu procedures midst of all plane flights. With this in mind, the authors attempted to develop limitations Software applications for mobile devices that will assist pilots in activities approaches. Developed software based on fuzzy controller, server and mobile applications. The article discusses the operation on fuzzy sets, which are included in the prepared concept of the software. Leaded also modeling of fuzzy set rules. The procedure components included in the artificial intelligence simulated in Matlab / Simulink. In addition, the results of the simulations carried out in the same the software. The whole summarizes the conclusions of the tests performed..

Autorzy:
dr inż. **Rafał Kowalik** – Wyższa Szkoła Oficerska Sił Powietrznych, Wydział Lotnictwa, Katedra Awioniki i Systemów Sterowania, r.kowalik@wsosp.pl.
inż. **Rafał Bieńczak** – Wyższa Szkoła Oficerska Sił Powietrznych, Wydział Lotnictwa, Katedra Awioniki i Systemów Sterowania, r.bienczak@wsosp.pl.
dr inż. **Andrzej Komorek** – Wyższa Szkoła Oficerska Sił Powietrznych, Wydział Lotnictwa, Katedra Awioniki i Systemów Sterowania, a.komorek@wsosp.pl.