Rozwiązania sprzętowe i programowe w sterowaniu robotami rehabilitacyjnymi Renus

Jacek Dunaj, Wojciech J. Klimasara

Przemysłowy Instytut Automatyki i Pomiarów PIAP

Streszczenie: W artykule opisano sposób realizacji sterowania oraz rozwiązań software'owych wykorzystanych podczas opracowywania dwóch robotów Renus-1 i Renus-2, przeznaczonych do wspomagania rehabilitacji ruchowej pacjentów po przebytych udarach mózgu lub ze schorzeniami ortopedycznymi. Sterowanie obu robotów wykonano na bazie komercyjnych elementów firmy Mitsubishi Electric stosowanych w automatyce przemysłowej. Ich wykorzystanie umożliwiło zbudowanie w pełni funkcjonalnych modeli obu robotów w stosunkowo krótkim czasie. W artykule przedstawiono strukturę układu sterowania i rozwiązania software'owe zastosowane przy opracowaniu oprogramowania obu robotów. Opisano interfejs użytkownika człowiek–robot zarówno dla pacjenta, jak i operatora-fizjoterapeuty. Opis interfejsu zilustrowano obrazami okien dialogowych. Roboty Renus-1 i Renus-2 opracowano i wykonano w Przemysłowym Instytucie Automatyki i Pomiarów PIAP w Warszawie.

Słowa kluczowe: rehabilitacja neurologiczna, robot rehabilitacyjny, sterownik PLC, aplikacja komputera PC, baza danych, sterowniki ODBC

DOI: 10.14313/PAR_214/100

1. Wprowadzenie

Rozwój robotów rehabilitacyjnych jest warunkowany sytuacją demograficzną, poziomem zamożności społeczeństwa, upowszechnieniem nowych technik wśród personelu medycznego, efektywnym i prostym systemem komunikacji, a także ceną. Ze względu na sytuację demograficzną przyszłość robotów rehabilitacyjnych jest jasna, ponieważ wraz z postępem medycyny coraz więcej ludzi dożywa sędziwego wieku. Urządzenia rehabilitacyjne mogą uczynić ich życie w miarę komfortowym. Warunkiem skuteczności rehabilitacji jest jej systematyczność i odpowiednia intensywność. Jednak ze względu na potencjalnie dużą liczbę pacjentów, do jej realizacji konwencjonalnymi metodami potrzeba wielu fizjoterapeutów.

Medycyna i inżynieria stoją przed koniecznością budowania robotów rehabilitacyjnych. Trzeba pamiętać, że styczność z nimi mają zazwyczaj ludzie starsi z różnymi ograniczeniami wynikającymi z chorób i wieku, którzy często nie mieli wcześniej kontaktu z tego typu urządzeniami. Szczególnie trudna jest rehabilitacja osób po przebytych udarach mózgu, ponieważ polega ona na długotrwałym, żmudnym „programowaniu" mózgu – rehabilitowana jest kończyna, ale pracuje się nad mózgiem. Dlatego robot rehabilitacyjny powinien być tak zbudowany, aby jego obsługa i wykonywanie za jego pomocą ćwiczeń nie było zbyt trudne, ponieważ działa to zniechęcająco i na potencjalnego pacjenta, i na operatora.

Robot rehabilitacyjny powinien gwarantować stworzenie określonych warunków dla operatora i dla pacjenta. Fizjoterapeuta musi mieć możliwość opracowania planu ćwiczeń, zaprogramowania robota, a następnie zainicjowania i nadzorowania realizacji ćwiczenia. U pacjenta natomiast trzeba przezwyciężyć obawę przed kontaktem z urządzeniem mechanicznym i uświadomić mu, że jest to proste w użyciu narzędzie. Ponadto urządzenie to powinno umożliwiać obiektywną ocenę postępów rehabilitacji.

Rehabilitacja neurologiczna składa się z dwóch faz: pasywnej (biernej) i czynnej (aktywnej). Z rehabilitacją pasywną mamy do czynienia, jeśli kończyna pacjenta jest zupełnie bezwładna. W takim przypadku robot wodzi tę kończynę po zadanej przez rehabilitanta trajektorii, a zadaniem pacjenta jest maksymalne koncentrowanie się na wykonywanym ćwiczeniu i „siłą woli" przeciwstawienie się robotowi, wykonując ruchy przeciwstawiające się. W tej fazie ćwiczeń tworzą się pierwsze zalążki neuronów, stanowiących o możliwości wykonywania ruchów. Rehabilitacja aktywna jest następnym etapem rehabilitacji – to pacjent ma odtwarzać przedstawioną mu trajektorię ruchu, wodząc w przestrzeni manipulatorem. Tutaj to robot stawia opór, a system ma zapewnić regulowanie siły oporu.

Celem projektu, prowadzonego w Przemysłowym Instytucie Automatyki i Pomiarów PIAP, było m.in. sprawdzenie czy na podstawie dostępnych na rynku środków (typowe i dostępne w handlu elementy stosowane w automatyce przemysłowej) można zbudować robota rehabilitacyjnego. W latach 2006–2010 oraz 2013–2014 opracowano, wykonano i oprogramowano w PIAP działające modele dwóch robotów rehabilitacyjnych: Renus-1 – dla kończyn górnych (rys. 1) oraz Renus-2 – dla kończyn dolnych (rys. 2). Umożliwiają one definiowanie i zapamiętywanie różnych trajektorii ruchu, a następnie realizację obu rodzajów rehabilitacji: czynnej i biernej.

Rys. 1. Robot Renus-1 do rehabilitacji kończyn górnych
Fig. 1. Renus-1 robot prototype for upper limbs rehabilitation

Rys. 2. Robot Renus-2 do rehabilitacji kończyn dolnych
Fig. 2. Renus-1 robot prototype for lower limbs rehabilitation

2. Zasada działania robotów rehabilitacyjnych Renus

Manipulatory robotów Renus mają po trzy stopnie swobody, co pozwala na:

- przemieszczenie uchwytu ręki w górę/w dół, w lewo/w prawo, do/od siebie – robot Renus-1 (fot. 3),
- przemieszczenie uchwytu stopy od/do pacjenta, jej skręcenie oraz zmianę pochylenia – robot Renus-2 (fot. 4).

Każdą z trzech osi manipulatora każdego z robotów porusza osobny silnik synchroniczny z magnesami trwałymi, sterowany za pomocą indywidualnego serwonapędu współpracującego z jednostką centralną sterownika. Jako urządzenia wykonawcze zastosowano sprzęt firmy Mitsubishi Electric. W obu zestawieniach zastosowano te same: jednostki centralne Q02HCPU i silniki synchroniczne HC-MFS43. Schemat sterowania silnikami elektrycznymi w przypadku obu robotów rehabilitacyjnych przedstawia się jak poniżej:

Robot Renus-1:

	→ moduł MR-J2S-20B →	silnik 1
Moduł Q02HCPU → moduł QD75M4	→ moduł MR-J2S-10B →	silnik 2
	→ moduł MR-J2S-10B →	silnik 3

Robot Renus-2:

	→ moduł MR-J3-40B →	silnik 1
Moduł Q02HCPU → moduł QD75MH4	→ moduł MR-J3-20B →	silnik 2
	→ moduł MR-J3-20B →	silnik 3

Jednostki centralne Q02HCPU oraz moduły MR-J2S--20B, MR-J2S-10B, MR-J3-40B, MR-J3-40B mają pamięci RAM o zawartości podtrzymywanej bateryjnie. Są w niej zapamiętane programy aplikacyjne (jednostki centralne) oraz parametry serwonapędów i położenia bazowe silników (moduły MR-J2S…, MR-J3…). Informacje te mogą zostać utracone w przypadku awarii/rozładowania baterii.

Każdy z silników napędzających manipulator ma stałe położenie bazowe (kąt obrotu wału silnika), względem którego odbywa się jego ruch. Przy pomocy serwonapędu sterownik może zadać położenie (kąt obrotu), do którego ma przemieścić się wał silnika, może także odczytać aktu-

Rys. 3. Ćwiczenie kończyny górnej wykonywane za pomocą robota Renus-1
Fig. 3. Upper limb rehabilitation exercises by the Renus-1 robot

Rys. 4. Ćwiczenie kończyny dolnej wykonywane przy pomocy robota Renus-2
Fig. 4. Lower limb rehabilitation exercises by the Renus-2 robot

Rys. 5. Wnętrze szafy sterowniczej robota Renus-2
Fig. 5. Inside of control cabinet of Renus-2 robot

alne położenie wału względem położenia bazowego. Wał silnika jest sprzężony z osią manipulatora robota za pomocą dodatkowych przekładni ruchu. Położenie uchwytu kończyny manipulatora robota w przestrzeni jest wypadkową położenia wałów (kątów obrotu względem pozycji bazowych) wszystkich trzech silników. Trajektorię ruchu określa zatem zbiór n położeń $[A_{1i}, A_{2i}, A_{3i}]$ gdzie: A_{1i} – określa kąt skręcenia wału silnika napędzającego oś nr 1 w punkcie i, A_{2i} – określa kąt skręcenia wału silnika napędzającego oś nr 2 w punkcie i, A_{3i} – określa kąt skręcenia wału silnika napędzającego oś nr 3 w punkcie i, gdzie $i \subset <1, 2,..,n>$.

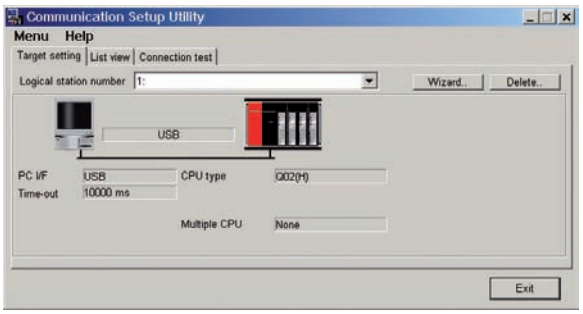
Definiowanie trajektorii ruchu polega na ręcznym przemieszczaniu uchwytu kończyny. W trakcie ruchu sterownik robota w stałych odstępach czasu odczytuje z każdego serwonapędu położenie wału silnika obsługiwanego przez ten serwonapęd, a następnie zapamiętuje odczytane położenia w pamięci.

Odtwarzanie trajektorii ruchu polega na czynności odwrotnej, tzn. wczytaniu do pamięci sterownika zdefiniowanej trajektorii, a następnie na doprowadzaniu wszystkich trzech silników do kolejnych zapamiętanych położeń. Roboty rehabilitacyjne Renus nie odtwarzają ruchu liniowego lub kołowego, tak jak robot przemysłowy, tylko przemieszczają manipulator „od położenia do położenia" wałów trzech silników synchronicznych.

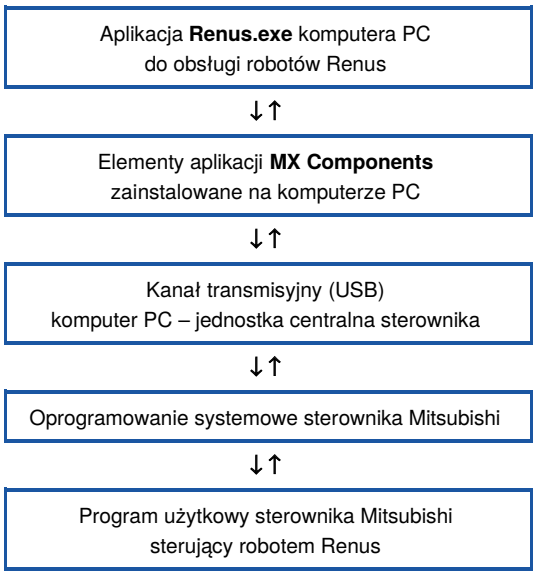
Robotów rehabilitacyjnych Renus nie wyposażono w panele sterowania typowe dla robotów przemysłowych. Rolę panelu może pełnić dowolny komputer PC z systemem operacyjnym Windows i oprogramowaniem do sterowania robotem. W takim rozwiązaniu nie są potrzebne specjalizowane karty, jedynym warunkiem jest wolny port USB.

3. Oprogramowanie systemowe i aplikacyjne sterowników Mitsubishi Q02HCPU robotów Renus

Programy aplikacyjne sterowników PLC zazwyczaj są tworzone za pomocą oprogramowania dostarczanego przez producenta sterownika. Oprogramowanie to umożliwia tworzenie i edycję programu aplikacyjnego oraz pełną obsługę



Rys. 6. Okno dialogowe aplikacji Communication Setup Utility do definiowania kanału transmisyjnego do/z sterownika Mitsubishi
Fig. 6. Dialog box of Communication Setup Utility application for define the transmission channel to/from the Mitsubishi controller



Rys. 7. Schemat wymiany informacji między aplikacjami sterownika robota Renus i komputera PC
Fig. 7. Diagramm of information exchange between the applications of Renus robot controller and PC computer

samego sterownika. Funkcje obsługi umożliwiają m.in. ładowanie programu aplikacyjnego do pamięci sterownika, jego uruchamianie, zatrzymywanie i debugging, podgląd stanu wejść, sterowanie wyjściami, podgląd i zmianę zawartości rejestrów pamięci, ustawianie zmiennych systemowych (np. daty i zegara sterownika), etc. Czynności te, w zależności od wyposażenia sterownika, są wykonywane za pośrednictwem standardowego interfejsu komunikacyjnego (RS-232, Ethernet, USB). Funkcje obsługi realizuje oprogramowanie systemowe sterownika i aplikacja do tworzenia programu PLC. W przypadku sterowników Mitsubishi taką aplikacją jest GX Developer, producent udostępnia też oprogramowanie MX Components, umożliwiające odwoływanie się do funkcji systemowych sterownika z poziomu dowolnego programu komputera.

Współpraca aplikacji do obsługi robota Renus i oprogramowania systemowego sterownika Mitsubiushi odbywa się wybranym kanałem transmisyjnym, który może wykorzystywać jeden z dostępnych w jednostce centralnej Q02HCPU

Oznaczenie rejestru	Opis
D100	Typ zastosowanego manipulatora: 1 – manipulator Renus-1, 2 – manipulator Renus-2
D101 **D102** **D103**	Aktualne położenia A_1, A_1, A_3 (kąty skręcenia) wału silnika napędzającego oś: nr 1, 2, 3 względem jego pozycji bazowej. Wartość ta nie jest podawana w jednostkach fizycznych (stopniach, radianach), tylko w inkrementach.
D104 **D105** **D106**	Aktualna wartość siły F_{A1}, F_{A2}, F_{A3} wywieranej na operatora przez manipulator lub na manipulator przez operatora przy obrocie osią odpowiednio nr 1, 2 i 3. Wartość ta nie jest podawana w jednostkach fizycznych (niutonach), tylko w inkrementach.
D107 **D108** **D109**	Numer błędu przy próbie obrotu osią odpowiednio nr 1, 2 i 3.
D110	Numer realizowanego punktu trajektorii (wartość liczona od 1)
D111	Liczba punktów trajektorii
D112	Rejestr flag bitowych i flag błędów.
D113 **D114** **D115**	Rejestry potwierdzenia komendy. Aplikacja komputera PC kod komendy do wykonania i jej potencjalne parametry wpisuje do trzech rejestrów D120, D121 i D122. Aplikacja sterownika robota potwierdza przyjęcie komendy do wykonania przez przepisanie zawartości D120 → D113, D121 → D114, D122 → D114.
D116–D119	Nieużywane, do przyszłego wykorzystania.

interfejsów: RS-232 lub USB. Obsługę transmisji wykonują zadeklarowane w projekcie aplikacji kontrolki Active-X instalowane wraz z aplikacją MX Components. Aby taka współpraca była możliwa, należy uprzednio za pomocą programu Communication Setup Utility zdefiniować kanał transmisyjny. Okno dialogowe tej aplikacji, będącej częścią oprogramowania MX Components, pokazano na rys. 6:

Definiowanie kanału polega na przyporządkowaniu mu numeru, określeniu rodzaju interfejsu oraz potencjalnych parametrów transmisji. Do tak zdefiniowanego kanału aplikacja komputera PC odwołuje się podając numer kanału jako parametr odpowiednich podprogramów bibliotecznych. Zaletą współpracy komputer–sterownik Mitsubishi jest to, że aplikację komputera można uruchamiać korzystając z innego typu sterownika niż docelowy, ponieważ oprogramowanie systemowe każdego sterownika Mitsubishi zapewnia ten sam zbiór funkcji obsługi. Chcąc zmienić rodzaj interfejsu, np. z USB na RS-232, wystarczy tylko za pomocą programu Communication Setup Utility zmienić deklarację interfejsu przyporządkowanego do kanału o danym numerze. Aplikacja komputera do obsługi robotów Renus korzysta z kanału o numerze 1 i interfejsu USB.

Działając według schematu (rys. 7) aplikacja komputera i aplikacja sterownika robota Renus mają dostęp do wspólnego obszaru pamięci sterownika podzielonego na kilkaset rejestrów, do/z których obie aplikacje mogą wpisywać i odczytywać informacje. Z poziomu aplikacji komputera można odczytywać lub ustawiać parametry systemowe sterownika i uruchamiać lub zatrzymać wykonywanie programu użytkowego. Funkcje te są przydatne podczas testowania kanału transmisyjnego między komputerem a układem sterowania robota Renus.

3.1. Rejestry stanu robota Renus

W pamięci sterownika Mitsubishi Q02HCPU zarezerwowano kilkadziesiąt rejestrów, tzw. rejestrów stanu, do których program PLC sterujący robotem Renus na bieżąco wpisuje różne informacje m.in. o aktualnym położeniu A_1, A_2, A_3 względem położeń bazowych wałów trzech silników poruszających osiami robota, o wartościach sił F_{A1}, F_{A2}, F_{A3} wywieranych na operatora przez manipulator lub na manipulator przez operatora, informacje o błędach, typie sterownika, etc. Wszystkie te rejestry mogą być odczytywane w czasie rzeczywistym przez aplikację komputera PC, a niektóre z nich – jak liczba punktów trajektorii – także zapisywane. Pełną listę tych rejestrów wraz z opisem przedstawiono w tabeli powyżej:

3.2. Rejestry komend robota Renus

Uruchomienie wykonania funkcji robota Renus jest inicjowane z poziomu aplikacji komputera obsługującej robota. Wysłanie komendy do robota polega na wstawieniu przez tę aplikację kodu komendy wraz z jej potencjalnymi parametrami do odpowiedniego rejestru (rejestrów) w przestrzeni adresowej sterownika PLC. Rejestry komend zajmują obszar od D120 do D122 w tej przestrzeni:

Oznaczenie rejestru	Opis
D120	kod komendy
D121, D122	potencjalne parametry komendy

Aktualnie roboty rehabilitacyjne Renus realizują wykonanie następujących komend:

Kod komendy (rejestr D120)	Opis komendy i parametry (rejestry D121, D122)
00	Zerowanie komend robota. Brak parametrów komendy.
01	Nauka robota (definiowanie trajektorii): [D121–D122] ⇔ [00–00]: [trajektoria prosta] – operator definiuje całą trajektorię realizowaną podczas rehabilitacji, od punktu początkowego do punktu końcowego i w tej postaci jest ona zapisywana w pamięci sterownika – operator ma do dyspozycji cały bufor do zapisu trajektorii. [D121–D122] ⇔ [00–01]: [trajektoria prosta] – wariant nauki dostępny tylko dla robota Renus-2 (kończyna dolna). Operator definiuje tylko połowę trajektorii ruchu realizowanej podczas rehabilitacji. Druga połowa trajektorii jest automatycznie generowana i zapamiętywana przez program sterownika robota jako ruch odbywający się wzdłuż linii zdefiniowanej przez operatora, ale realizowany w odwrotnej kolejności punktów. Operator ma do dyspozycji połowę pojemności bufora przeznaczonego do zapisu trajektorii. [D121–D122] ⇔ [01–00]: [trajektoria odwrócona] – wariant nauki dostępny tylko dla robota Renus-2 (kończyna dolna). Operator definiuje całą trajektorię realizowaną podczas rehabilitacji, od punktu początkowego do punktu końcowego. Po zakończeniu nauki, zapamiętane położenia wału silnika powodującego skręcanie stopy pacjenta są „odwracane" względem jego położenia bazowego, skręcenie w lewą stronę będzie teraz skręceniem w prawą stronę i vice-versa. Operator ma do dyspozycji całą pojemność bufora przeznaczonego do zapisu trajektorii.
02–05	Kody komend nauki robota (definiowania trajektorii) zarezerwowane do przyszłego wykorzystania.
06	Wykonywanie rehabilitacji biernej, polegającej na odtwarzaniu przez manipulator robota zadanej trajektorii ruchu. Brak parametrów komendy.
07	Wykonywanie rehabilitacji aktywnej, polegającej na próbie odtwarzania przez pacjenta zadanej trajektorii ruchu. Brak parametrów komendy.
08	Przepisanie trajektorii z komputera PC do sterownika PLC robota Renus. Brak parametrów komendy.
09–10	Kody komend zarezerwowane do przyszłego wykorzystania.
11	[D121–D122] ⇔ [Prędkość–Siła] Ustawienie prędkości i siły realizowanej przez manipulator. Oba parametry, podawane w procentach od 0 %, do 100 %, odległości między minimalną a maksymalną wartością (0 % oznacza prędkość minimalną, 100 % oznacza prędkość maksymalną, analogicznie dla siły).

3.3. Rejestry trajektorii robota

Trajektoria robota rehabilitacyjnego może być za pomocą aplikacji odczytywana ze sterownika robota (funkcja uczenia), jak też przesyłana do sterownika (realizacja rehablitacji aktywnej i biernej). Definicja każdego punktu trajektorii zawiera informację o położeniach A_1, A_2 i A_3 względem położeń bazowych wałów silników poruszających osie robota oraz informacje o wartościach sił F_{A1}, F_{A2} i F_{A3} wywieranych na operatora przez manipulator lub na manipulator przez operatora przy poruszaniu osiami silników. Wartości położeń i sił nie są podawane w jednostkach fizycznych, tylko inkrementach. Program aplikacyjny sterownika Mitsubishi Q02HCPU robota zapewnia możliwość definiowania trajektorii złożonej maksymalnie z 600 punktów. Informacje o poszczególnych punktach trajektorii są przechowywane w pamięci sterownika w następujących rejestrach:

Położenia A_1, A_2, A_3 wałów silników robota

Numer punktu	Położenie A_1	Położenie A_2	Położenie A_3
1	D1000	D1001	D1002
2	D1003	D1004	D1005
…	…	…	…
599	D2794	D2795	D2796
600	D2797	D2798	D2799

Wartości sił F_{A1}, F_{A2} i F_{A3}

Numer punktu	Siła F_{A1}	Siła F_{A2}	Siła F_{A3}
1	D2800	D2801	D2802
2	D2803	D2804	D2805
…	…	…	…
599	D3594	D3595	D3596
600	D3597	D3598	D3599

Definicja trajektorii obejmuje także informację o liczbie punktów trajektorii pamiętaną w rejestrze **D111** sterownika.

4. Oprogramowanie sterujące robotami rehabilitacyjnymi Renus

Roboty rehabilitacyjne Renus nie są urządzeniami w pełni autonomicznymi, ponieważ do ich funkcjonowania niezbędny jest komputer z systemem operacyjnym Windows i odpowiednim oprogramowaniem. Oprócz pakietu MX Components firmy Mitsubishi Electric w skład niezbędnego oprogramowania wchodzą:

- aplikacja Renus.exe wykorzystująca pakiet MX Components, która bezpośrednio steruje danym robotem, zarządza bazami danych związanymi z wykonywaniem zadań rehabilitacji i pełni rolę panelu operatorskiego,
- tekstowy plik konfiguracyjny Renus.ini, w którym są zawarte informacje modyfikujące działanie aplikacji Renus.exe,
- baza danych pacjentów zapisana w pliku RenusBazaDanychPacjentow.mdb, do przechowywania informacji o pacjentach rehabilitowanych za pomocą robotów Renus,
- baza danych trajektorii zapisana w pliku RenusBazaDanychTrajektorii.mdb – do przechowywania informacji o zdefiniowanych trajektoriach robotów Renus.

4.1. Aplikacja komputera do obsługi robotów Renus

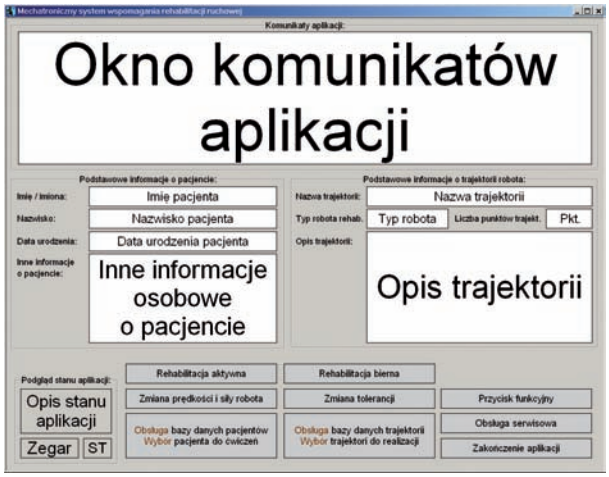
Aplikacja Renus.exe została utworzona w języku C++ i uruchomiona w środowisku Visual Studio .NET 2003. Nie wymaga ona procesu instalacji, wystarczy tylko ją skopiować do pamięci masowej komputera. Dodatkowo do jej uruchomiona wymagana jest obecność dwóch bibliotek mfc71.dll i msvcr71.dll dostępnych w środowisku Visual Studio .NET. Konieczne jest także zainstalowanie aplikacji MX Components.

Główne okno dialogowe aplikacji Renus.exe (rys. 8) składa się z dwunastu okienek informacyjnych i dziewięciu przycisków funkcyjnych. Najważniejsze z nich to:

- **Okno komunikatów aplikacji** – tu są wyświetlane komunikaty dla operatora robota rehabilitacyjnego,
- **Podstawowe informacje o pacjencie** – informacje odczytane z rekordu bazy danych pacjentów,
- **Podstawowe informacje o trajektorii robota** – niektóre informacje odczytane z rekordu bazy danych trajektorii,
- **Podgląd stanu aplikacji** – informacja o charakterze serwisowym – wskazanie zegara komputera oraz opis i numer etapu czynności wykonywanej przez aplikację.
- **Rehabilitacja aktywna** – uruchamia czynności związane z obsługą rehabilitacji aktywnej,
- **Rehabilitacja bierna** – uruchamia czynności związane z obsługą rehabilitacji biernej,
- **Zmiana prędkości i siły robota** – uruchamia czynności związane z obsługą zmiany nastaw siły (rehabilitacja aktywna), z jaką manipulator oddziaływuje na pacjenta i prędkości (rehabilitacja bierna), z jaką manipulator realizuje trajektorię ruchu,
- **Zmiana tolerancji** – uruchamia czynności związane ze zmianami nastaw tolerancji doprowadzenia każdego z silników do kolejnych położeń wyznaczających trajektorię realizowaną podczas rehabilitacji aktywnej,
- **Obsługa bazy danych pacjentów/Wybór pacjenta do ćwiczeń** – uruchamia czynności związane z obsługą bazy danych pacjentów, pozwala na wybór pacjenta, który będzie wykonywał ćwiczenie rehabilitacyjne z użyciem robota Renus,
- **Obsługa bazy danych trajektorii/Wybór trajektorii do realizacji** – uruchamia czynności związane z obsługą bazy danych trajektorii, w tym funkcję nauki robota, czyli definiowanie trajektorii, pozwala także na wskazanie trajektorii, która będzie realizowana podczas wykonywania rehabilitacji aktywnej i biernej,
- **Przycisk funkcyjny** – przycisk uniwersalny, którego znaczenie zależy od czynności aktualnie wykonywanej przez aplikację (zmienia się opis funkcji wyświetlanej na przycisku); zwykle służy do potwierdzania przez operatora komunikatów o błędach,
- **Obsługa serwisowa** – uruchamia czynności związane z obsługą serwisową jednostki centralnej sterownika robota,
- **Zakończenie aplikacji** – kończy wykonywanie aplikacji Renus.exe.

4.2. Plik konfiguracyjny aplikacji do obsługi robotów Renus

Roboty rehabilitacyjne Renus to jeszcze nie urządzenia komercyjne i nadal są poddawane różnym modyfikacjom. Aby uprościć sposób parametryzacji niektórych zmiennych mających wpływ na działanie aplikacji Renus.exe, wprowadzono dodatkowy, tekstowy plik konfiguracyjny Renus.ini pamiętany w tym samym folderze co aplikacja. Każdy wiersz tego pliku jest oddzielnie analizowany przez aplikację. Jeśli nie zawiera znaku równości „=” to jest przez nią traktowany jako komentarz. Jeśli jednak aplikacja wykryje w wierszu znak równości to wszystkie znaki od początku wiersza do pierwszego znaku „=” są traktowane jako nazwa zmiennej, a wszystkie znaki na prawo od tego



Rys. 8. Główne okno dialogowe aplikacji Renus.exe
Fig. 8. The main dialog box of Renus.exe application

znaku – jako wartość zmiennej. Jeśli dana zmienna nie wystąpi w pliku konfiguracyjnym to aplikacja przyjmuje jej wartość domyślną. Przy ustalaniu nazw zmiennych pliku konfiguracyjnego przyjęto zasadę, że nazwy te odpowiadają nazwom zmiennych globalnych zdefiniowanych w plikach źródłowych aplikacji Renus.exe.

Zmienne pliku konfiguracyjnego pozwalają ustalać m.in.:

- wartości niektórych stałych czasowych stosowanych w aplikacji, w tym czasy wyświetlania komunikatów informacyjnych aplikacji,
- konfigurację sprzętową dla aplikacji Renus.exe: w tym typ jednostki centralnej sterownika Mitsubishi, i numer kanału transmisyjnego do obsługi transmisji komputer PC–sterownik,
- położenia i długości pól pamięci sterownika, gdzie przechowywane są informacje o stanie robota i współrzędne punktów określających trajektorię,
- informacje związane z konfigurowaniem dostępu aplikacji Renus.exe do bazy danych pacjentów i bazy danych trajektorii,
- informacje o położeniu plików tekstowych z definicjami trajektorii,
- maksymalne dopuszczalne wartości położeń i wartości sił wywieranych na osie silników robotów rehabilitacyjnych.

Ostatnia grupa zmiennych ma szczególne znaczenie. Każdą z trzech osi manipulatora robota (Renus-1, Renus-2) porusza osobny silnik synchroniczny, sterowany indywidualnym serwonapędem. Serwonapęd umożliwia także odczyt aktualnej pozycji wału silnika (kąta obrotu) mierzonej względem jego pozycji bazowej. Odczyt taki można wykonywać podczas ruchu zadanego przez sterownik oraz podczas obrotu ręcznie wymuszonego przez operatora. Teoretyczny zakres ruchu silnika poza robotem jest znacznie większy niż roboczy zakres ruchu tego samego silnika zamontowanego w robocie, ponieważ w tym drugim przypadku jest on ograniczony mechaniczną konstrukcją samego manipulatora. Możliwe jest także przeprogramowanie bazowej pozycji każdego silnika, przez co realizacja trajektorii zdefiniowanej przy innej pozycji bazowej może powodować kolizję z ograniczeniami mechanicznymi. Aby uniknąć problemów, aplikacja Renus.exe umożliwia parametryzowanie dopuszczalnych zakresów ruchu każdej osi i sił wymuszających taki ruch lub sił, z jakimi robot przeciwstawia się ręcznemu wymuszeniu obrotu. Parametryzacja ta jest indywidualnie wykonywana dla robota Renus-1 i Renus-2, a ustalone przy jej pomocy wartości zmiennych służą do weryfikacji trajektorii ruchu oraz pozwalają odpowiednio skalować wykresy zmian położeń i sił w funkcji czasu wyświetlane w różnych oknach dialogowych aplikacji.

4.3. Bazy danych wykorzystywane podczas pracy z robotami Renus

Aplikacja Renus.exe współpracuje z dwiema bazami danych: bazą danych pacjentów oraz bazą danych trajektorii. W obu przypadkach funkcje obsługi tych baz są realizowane za pomocą sterowników ODBC (ang. Open Data Base Connectivity), które są integralną częścią systemu operacyjnego Windows. Pozwalają one na wykonywanie operacji na zawartości bazy bez względu na jej format (Access, Excel, dBase, Paradox, etc.) pod warunkiem zachowania nazw tabel oraz nazw i charakteru pól zawartych w poszczególnych tabelach. Ze względu na potencjalnie najłatwiejszą dostępność obie bazy danych utworzono w formacie Access 2000 i zapisano w plikach:

- RenusBazaDanychPacjentow.mdb
- RenusBazaDanychTrajektorii.mdb

Miejsce posadowienia każdej z baz może być dowolne, ponieważ jest ono określane podczas konfigurowania źródeł danych ODBC przy pomocy narzędzi dostępnych w systemie Windows. Bazy można zatem osadzić na innym komputerze niż ten bezpośrednio obsługujący robota rehabilitacyjnego, o ile będą się one znajdować na udostępnionym dysku sieciowym. Komputer przeznaczony do obsługi robota Renus nie musi mieć zainstalowanego programu Access, ponieważ aplikacja Renus.exe zapewnia podstawowe funkcje związane z obsługą obu tych baz:

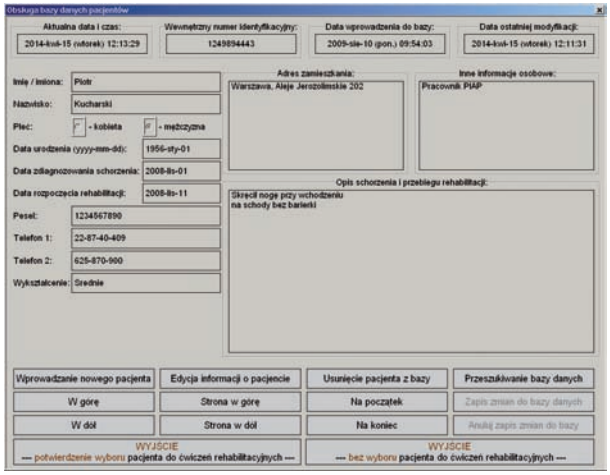
- wprowadzanie nowego rekordu do bazy,
- modyfikacja zawartości rekordu uprzednio wprowadzonego do bazy,
- usuwanie rekordu z bazy danych,
- przeglądanie zawartości bazy danych.

Aplikacja nie pozwala jednak na tworzenie kwerend, raportów ani wydruków związanych z wybraną bazą. Do tego celu trzeba wykorzystać program Access.

Obie bazy zawierają taki sam zbiór trzech pól wypełnianych automatycznie przez aplikację Renus.exe podczas definiowania nowego rekordu lub modyfikacji rekordu uprzednio wprowadzonego do bazy. Polami tymi są:

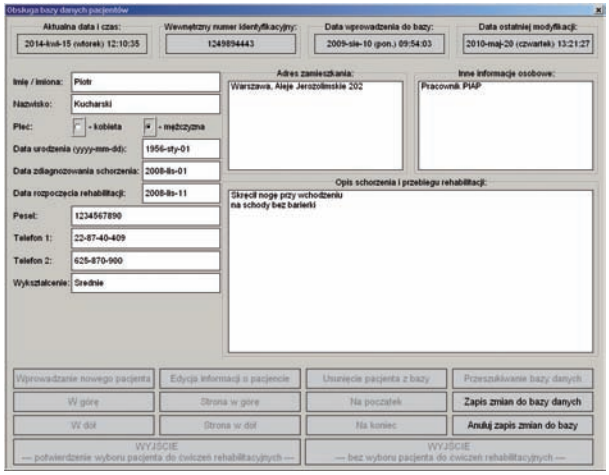
Wewnętrzny numer identyfikacyjny: Informacja zawarta w tym polu fizycznie odpowiada liczbie sekund, jakie upłynęły od dnia 1970-01-01 od godz 0:00 (wartość tę oblicza jedna z dostępnych w Visual Studio .NET procedur bibliotecznych). Pole to zostało wskazane jako tzw. klucz podstawowy bazy danych, ponieważ jako klucz podstawowy powinno się zawsze wskazywać takie pole, w którym informacja zawarta w poszczególnych rekordach jest unikatowa.

Data wprowadzenia do bazy danych: Zawartość tego pola pozwala ustalić dokładną datę (rok, miesiąc, dzień, godzina, minuta, sekunda) wprowadzenia do bazy danego rekordu.



Rys. 9. Okno dialogowe obsługi bazy danych pacjentów w trybie przeglądania bazy

Fig. 9. Dialog box of patients database maintenance in the review mode



Rys. 10. Okno dialogowe obsługi bazy danych pacjentów w trybie wprowadzania lub modyfikacji rekordu

Fig. 10. Dialog box of patients database maintenance in the input mode or modify the records

Data ostatniej modyfikacji: Zawartość tego pola pozwala ustalić dokładną datę (rok, miesiąc, dzień, godzina, minuta, sekunda) ostatniej modyfikacji danego rekordu.

4.3.1. Baza danych pacjentów

Bazę danych pacjentów można stosować do gromadzenia informacji o osobach rehabilitowanych za pomocą robotów Renus lub na jej podstawie utworzyć inną bazę w formacie wygodnym dla ośrodka korzystającego z tych robotów. W obecnej wersji oprogramowania baza ta ma charakter przykładowy i nie jest powiązana z czynnościami wykonywanymi przy pomocy robotów Renus. Zawiera ona pojedynczą tabelę RenusTabelaDanychPacjentow, w której przechowywane są informacje o pacjencie – imię, nazwisko, datę urodzenia, płeć, pesel, adres zamieszkania, wykształcenie, telefony kontaktowe, inne informacje osobowe, daty: zdiagnozowania schorzenia i rozpoczęcia rehabilitacji, opis schorzenia.

Obsługę bazy danych pacjentów można wykonywać w dwóch trybach pracy:

- w trybie przeglądania zawartości bazy (rys. 9), gdzie możliwe jest usuwanie rekordów (usuwanie pacjenta z bazy),
- w trybie wprowadzania i edycji rekordu (rys. 10), a więc wprowadzania do bazy nowego pacjenta lub modyfikacji informacji o pacjencie już wspisanym do bazy.

W zależności od wybranego trybu pracy część przycisków funkcyjnych okienka dialogowego pozostaje aktywna (czarna czcionka komentarzy) lub zablokowana (ciemnoszara czcionka komentarzy). Przejście z trybu przeglądania do trybu edycji następuje po wybraniu przycisku „Wprowadzanie nowego pacjenta" lub „Edycja informacji o pacjencie", przejście w odwrotnym kierunku – po wybraniu przycisku „Zapis zmian do bazy" lub „Anuluj zapis zmian do bazy".

4.3.2. Baza danych trajektorii

Aby było możliwe wykorzystanie robota Renus do prowadzenia ćwiczeń rehabilitacyjnych oprogramowanie musi zapewnić przechowywanie informacji o trajektorii ruchu manipulatora. Zrealizowano to w następujący sposób:
W pliku konfiguracyjnym Renus.ini aplikacji zdefiniowano zmienną o nazwie: FolderDefinicjeTrajektorii

Jej wartość określa miejsce (urządzenie i katalog), w którym przechowywane są pliki tekstowe z definicjami trajektorii. Każdy taki plik zawiera informację o pojedynczej trajektorii. Trajektoria robota Renus składa się z kilkudziesięciu punktów, między którymi realizowany jest ruch manipulatora. Z powodu ograniczeń programowych trajektoria może zawierać nie mniej niż 4 i nie więcej niż 600 punktów. Każdy punkt trajektorii określa sześć liczb, z których pierwsze trzy to położenia osi A_1, A_2 i A_3 silników manipulatora względem ich położeń bazowych, a następne trzy to wartości sił F_{A1}, F_{A2} i F_{A3}, jakie operator ma wywierać na osie silników lub silniki będą wywierać na operatora w trakcie realizacji ruchu do danego punktu trajektorii.

Aplikacja Renus.exe korzysta z bazy danych trajektorii zawierającej następujące informacje: nazwę trajektorii, datę wprowadzenia trajektorii do bazy danych i datę wykonania jej ostatniej modyfikacji, liczbę punktów z których składa się dana trajektoria, typ robota rehabilitacyjnego, na którym można realizować daną trajektorię, nazwę pliku z definicją trajektorii.

Takie „dwustopniowe" przechowywanie informacji o trajektoriach (baza danych + pliki tekstowe) jest spowodowane ograniczeniami samej bazy. W skrajnym przypadku trajektoria może składać się z 600 punktów, a zatem do jej opisania potrzeba $6 \cdot 600 = 3600$ liczb. Pojedyncza tabela bazy danych w formacie Access może zawierać ok. 250 pól (kolumn), zatem do zapamiętania całej trajektorii, złożonej z 600 punktów, trzeba byłoby zdefiniować bazę zawierającą co najmniej 15 tabel. Obsługa takiej bazy z poziomu aplikacji Renus.exe byłaby możliwa, ale bardzo nieefektywna. Baza danych trajektorii zawiera pojedynczą tabelę RenusTabelaDanychTrajektorii z ośmioma polami, z których tylko dwa wypełnia operator robota, a pozostałe pięć automatycznie uzupełnia aplikacja Renus.exe. Pola wypełniane przez operatora to:

Nazwa trajektorii – pole przeznaczone do przechowywania nazwy trajektorii, m.in. na jej podstawie aplikacja Renus.exe automatycznie generuje nazwę pliku tekstowego ze współrzędnymi punktów trajektorii.

Opis trajektorii – pole przeznaczone do przechowywania krótkiego opisu trajektorii.

Wartości kolejnych dwóch pól aplikacja Renus.exe uzupełnia na podstawie odczytów ze sterownika Mitsubishi Q02HCPU po zakończeniu nauki robota:

Typ robota rehabilitacyjnego: pole zawierające informację o typie robota rehabilitacyjnego. Typ odczytany z rejestru D100 sterownika. Wartość wpisana do tego pola służy do późniejszej weryfikacji robota, na którym ma być zealizowana trajektoria. Ma to zapobiec próbie realizacji trajektorii na robocie innego typu niż ten, na którym zdefiniowano trajektorię.

Liczba punktow trajektorii: pole, którego zawartość określa liczbę punktów trajektorii. Wartość tę aplikacja odczytuje z rejestru D111 sterownika i służy ona do późniejszej weryfikacji informacji zawartej w pliku z definicją trajektorii (liczba wierszy w pliku musi się zgadzać z liczbą punktów trajektorii).

Ostatnie pole bazy wypełniane przez aplikację Renus.exe zawiera nazwę pliku tekstowego ze współrzędnymi kolejnych punktów trajektorii:

Nazwa pliku z punktami trajektorii pole, w którym pamiętana jest nazwa pliku tekstowego z definicją trajektorii. Nazwę folderu z tym plikiem określa wartość przypisana do zmiennej „FolderDefinicjeTrajektorii" w pliku konfiguracyjnym aplikacji. Nazwę pliku z definicją trajektorii aplikacja generuje na podstawie daty wprowadzenia trajektorii do bazy danych oraz zawartości pola „Nazwa trajektorii". Jest to wykonywane w następujący sposób:

Pierwsze 21 znaków składających się na nazwę pliku stanowi ciąg znakowy uporządkowany według następującego schematu:

yyyy-mm-dd_hh-mi-ss___

gdzie:

yyyy – to cztery cyfry dziesiętne określające ***rok***,
mm – to dwie cyfry dziesiętne określające ***miesiąc***,
dd – to dwie cyfry dziesiętne określające ***dzień***,
hh – to dwie cyfry dziesiętne określające ***godzinę***,
mi – to dwie cyfry dziesiętne określające ***minutę***,
ss – to dwie cyfry dziesiętne określające ***sekundę***

Ciąg ten określa datę wprowadzenia trajektorii do bazy danych i jest zakończony dwoma znakami „_" (ang. **underscore**). Kolejne znaki składające się na nazwę pliku są dopisywane do tej nazwy na podstawie zawartości pola bazy „Nazwa trajektorii", przy czym wszystkie znaki **\:/*?<>|**, które nie mogą występować w nazwie pliku są zastępowane znakami znakami „_". Rozszerzenie nazwy pliku z definicją trajektorii jest stałe i ma postać .TRA.

4.3.3. Plik tekstowy do zapisu definicji trajektorii

Definicja trajektorii jest zapisywana w pliku tekstowym. Każdy wiersz pliku zawiera informację o jednym punkcie trajektorii i składa się z sześciu grup cyfr hexadecymalnych, po cztery cyfry w grupie tworząc 24-znakowy ciąg. Kolejne grupy cyfr zawierają informacje o położeniach A_1, A_2 i A_3 względem pozycji bazowych osi silników manipulatora i wartościach sił F_{A1}, F_{A2} i F_{A3} wywieranych przez operatora na osie silników lub na operatora przez osie silników podczas realizacji ruchu do danego punktu. Liczba wierszy w pliku odpowiada liczbie punktów trajektorii zapisanych w polu „Liczba punktow trajektorii" bazy danych (jest to jedno z kryterium weryfikacji poprawności danych). Przykład zapisu definiującego 5-punktową trajektorię ruchu ma postać:

```
000011621004000000000000
0083101E0FF60083001E00F6
020A0D300E91000A00300091
04230A190BA40023001900A4
06410B650B00004100650000
```

Aplikacja odczytuje trajektorię robota z rejestrów D1000—D2799 (wartości położeń) i D2800—D3599 (wartości sił) sterownika Mitsubishi Q02HCPU po zakończeniu nauki robota.

4.3.4. Obsługa bazy danych trajektorii z poziomu aplikacji Renus.exe

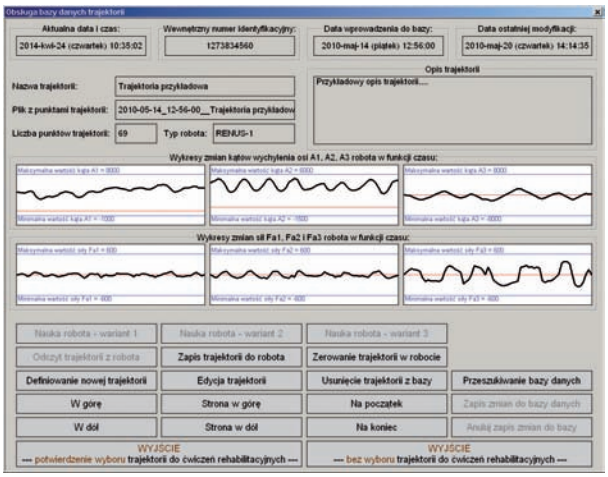
W zakresie obsługi bazy danych trajektorii (rys. 11 i 12) aplikacja umożliwia przeglądanie informacji zawartych w bazie danych trajektorii wraz z graficzną prezentacją całej trajektorii w postaci sześciu wykresów położeń A_1, A_2 i A_3 oraz sił F_{A1}, F_{A2} i F_{A3} w funkcji czasu, wprowadzanie do bazy nowej trajektorii, edycja lub usunięnie z bazy trajektorii uprzednio zdefiniowanej oraz wybór trajektorii do realizacji podczas rehabilitacji czynnej lub biernej.

Obsługę bazy danych trajektorii można realizować na dwa sposoby:

– w trybie przeglądania zawartości bazy (rys. 11), gdzie można też usuwać rekordy opisujące trajektorie,
– w trybie wprowadzania i edycji rekordu (rys. 12) opisującego trajektorię. W tym trybie obsługi można wykonać także funkcję „nauki robota".

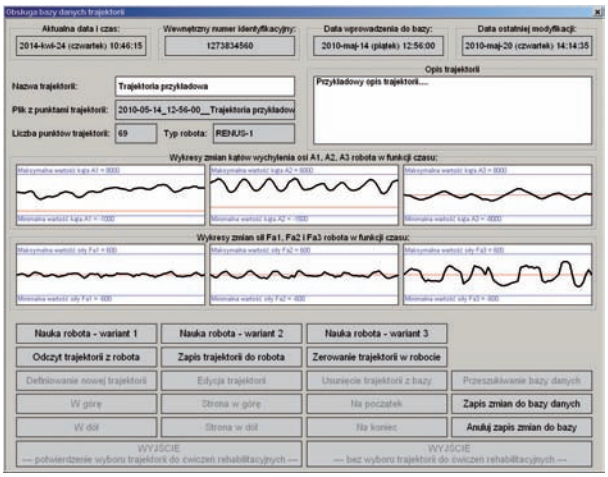
W zależności od wybranego trybu pracy część przycisków funkcyjnych okienka dialogowego pozostaje aktywna (czarna czcionka komentarzy) lub zablokowana (ciemnoszara czcionka komentarzy). Przejście z trybu przeglądania do trybu edycji następuje po wybraniu przycisku „Definiowanie nowej trajektorii" lub „Edycja trajektori", przejście w odwrotnym kierunku – po wybraniu przycisku „Zapis zmian do bazy" lub „Anuluj zapis zmian do bazy".

W trybie przeglądania zawartości bazy informacja zawarta w każdym odczytanym rekordzie jest dodatkowo weryfikowana. Sprawdzany jest typ robota, obecność pliku tekstowego z zapisem trajektorii i jego format. Aplikacja porównuje także liczbę wierszy w pliku z informacją znajdującą się w polu „Liczba punktow trajektorii" rekordu bazy, a następnie weryfikuje, czy współrzędne wszystkich punktów trajektori A_1, A_2 i A_3 oraz wartości sił F_{A1}, F_{A2} i F_{A3} mieszczą się w dopuszczalnych przedziałach określonych przez wartości przypisane do odpowiednich zmiennych w pliku konfiguracyjnym aplikacji. Pierwszy wykryty błąd dyskwalifikuje trajektorię, powodując wyświetlenie stosownego komunikatu i przerwanie dalszej weryfikacji. W trybie przeglądania bazy ważną rolę pełnią dwa najniższe przyciski okna dialogowego opisane komentarzami WYJŚCIE. Oprócz funkcji związanej



Rys. 11. Okno dialogowe obsługi bazy danych trajektorii w trybie przeglądania bazy

Fig. 11. Dialog box of trajectory database maintenance in the review mode



Rys. 12. Okno dialogowe obsługi bazy danych trajektorii w trybie wprowadzania lub modyfikacji rekordu

Fig. 12. Dialog box of trajectory database maintenance in the input mode or modify the records

z zamknięciem okna dialogowego powodują one potwierdzenie lub anulowanie wyboru trajektorii zdefiniowanej w aktualnym rekordzie bazy do realizacji podczas przeprowadzania rehabilitacji aktywnej lub biernej. Wyjście z potwierdzeniem wyboru możliwe jest tylko dla trajektorii, która pomyślnie przeszła test weryfikacji.

5. Nauka robota

W obecnej wersji oprogramowania robotów Renus nauka jest jedynym sposobem definiowania trajektorii ruchu manipulatora. Polega ona na manualnym przemieszczaniu manipulatora w przestrzeni i zapisywaniu w pamięci sterownika co 1 sekundę jego pozycji. Pozycję manipulatora w każdym punkcie wyznaczają położenia A_1, A_2 i A_3 osi trzech silników robota odniesione do ich pozycji bazowych oraz wartości trzech sił F_{A1}, F_{A2}, F_{A3} wywieranych przez operatora na osie tych silników podczas definiowania trajektorii.

Aplikacja Renus.exe umożliwia definiowanie nowej trajektorii lub edycję trajektorii już zdefiniowanej. Edycja trajektorii może jednak dotyczyć tylko jej nazwy i opisu, ale nie przebiegu. Obecnie edycja samego przebiegu nie jest możliwa, gdyż nie ma możliwości „przejęcia" współrzędnych części punktów z poprzedniej wersji. Edycja przebiegu może polegać tylko na jego ponownym zdefiniowaniu od początku do końca. Należy jednak zaznaczyć, że w przebieg trajektorii można ingerować, ale tylko poprzez ręcznie wprowadzanie poprawek do bazy danych trajektorii i edycję pliku tekstowego z jej zapisem, co nie jest, naturalnie, zalecane.

Funkcję nauki robota można uruchamiać w trybie wprowadzania nowego rekordu (nowej trajektorii) lub modyfikacji istniejącego rekordu bazy danych (rys. 13 i 14). w jednym z trzech wariantów:

Wariant 1 nauki jest stosowany w przypadku obu robotów Renus-1 i Renus-2. Trajektoria zdefiniowana w tym wariancie nie podlega dodatkowym modyfikacjom wprowadzanym przez oprogramowanie sterownika Mitsubishi Q02HCPU. Trajektoria może składać się maksymalnie z 600 punktów.

W wariancie 2 nauki, dostępnym dla robota Renus-2, operator definiuje tylko pierwszą połowę trajektorii, może więc zdefiniować maksymalnie 300 punktów. Druga część trajektorii jest generowana automatycznie przez oprogramowanie sterownika jako powrót po tej samej drodze.

W wariancie 3 nauki, dostępnym tylko dla robota Renus-2, operator definiuje całą trajektorię realizowaną podczas rehabilitacji, od punktu początkowego do punktu końcowego, czyli może zdefiniować maksymalnie 600 punktów. Po zakończeniu nauki, zapamiętane położenia wału silnika powodującego skręcanie stopy pacjenta są „odwracane" względem jego położenia bazowego, skręcenie w lewą stronę będzie teraz skręceniem w prawą stronę i vice-versa.

Tylko robot Renus-2 do rehabilitacji kończyny dolnej, umożliwia obsługę wszystkich trzech wariantów nauki. Wynika to ze specyfiki ćwiczeń osób po udarach, kiedy upośledzeniu najczęściej ulega tylko jedna połowa ciała. W takim przypadku trajektorię do ćwiczeń można zaprogramować za pomocą zdrowej kończyny pacjenta i zapamiętać ją jako lustrzane odbicie dla rehabilitacji kończyny upośledzonej.

Przeglądanie i edycję części informacji zawartej w bazie danych trajektorii można wykonywać bez udziału robota rehabilitacyjnego, dlatego przy otwieraniu okna dialogowego do obsługi tej bazy nie są wykonywane żadne czynności diagnostyczne związane ze sprawdzeniem kanału transmisji między komputerem PC a sterownikiem robota. Jeśli jednak operator zamierza przeprowadzić naukę robota, to muszą być spełnione dwa warunki:

– komputer PC i sterownik robota powinny być sprzężone sprawnym kanałem komunikacyjnym umożliwiającym przesyłanie komend, odczyt rejestrów stanu robota oraz transmisję trajektorii z robota,
– wybrany wariant nauki możliwy jest do realizacji przy pomocy danego robota (robota Renus-1 można „uczyć" tylko w wariancie I).

Warunki te są sprawdzane bezpośrednio po wyświetleniu okna dialogowego do nauki robota. W trakcie wykonywania tych czynności, a także podczas bezpośredniego rejestrowa-

Rys. 13. Okno dialogowe nauki robota wyświetlane podczas przerwy w rejestacji trajektorii

Fig. 13. The dialog box of robot learning that is shown during the break of trajectory registration

Rys. 14. Okno dialogowe nauki robota wyświetlane podczas rejestacji trajektorii

Fig. 14. The dialog box of robot learning that is shown during the trajectory registration

nia trajektorii, aplikacja Renus.exe monitoruje kanał transmisyjny oraz sprawdza kody błędów odczytane z rejestrów stanu robota. Każdy błąd jest błędem krytycznym, powodującym wyświetlenie komunikatu z opisem błędu i zakończenie nauki. Przed rozpoczęciem nauki i po jej zakończeniu manipulator robota jest bazowany, tzn. osie każdego z trzech silników napędzających manipulator są doprowadzane do ich pozycji bazowych.

Okna dialogowe nauki robota zawierają informacje o typie robota rehabilitacyjnego, liczbie zdefiniowanych punktów trajektorii, stanie nożnych przycisków sprzętowych podłączonych do sterownika, a także aktualne wartości położeń (kąty A_1, A_2, A_3) osi trzech silników manipulatora robota, i aktualne wartości sił (F_{A1}, F_{A2} i F_{A3}), z jakimi operator oddziaływuje na osie każdego z tych silników. Stan przycisków, wartości położeń i sił są co 0,1 s odczytywane z rejestrów stanu sterownika robota i uaktualniane na ekranie. Dodatkowo w sześciu okienkach aplikacja wyświetla przebiegi położeń A_1, A_2, A_3 i sił F_{A1}, F_{A2} i F_{A3} w funkcji czasu w taki sposób, że kolejny pixel wykresu odpowiada kolejnemu odczytowi położenia lub siły. W przypadku, gdy liczba punktów odczytu przekracza szerokość okienka (w pixelach), wykresy są przesuwane o jeden pixel w lewo zwalniając miejsce na kolejny odczyt. Wykresy te są jednak uaktualniane tylko podczas bezpośredniej rejestracji trajektorii.

Aby rozpocząć rejestrację trajektorii operator powinien wcisnąć czarny, sprzętowy przycisk nożny, powodując zwolnienie hamulców i zmianę tła na żółte większości okienek pola dialogowego nauki robota (patrz fot. 13 i 14), a następnie przemieszczać manipulator w przestrzeni. Wykresy zmian położenia osi silników i sił wywieranych na nie przez operatora pełnią tutaj ważną rolę. Operator powinien tak przemieszczać manipulator, aby:

– żaden z wykresów położenia A_1, A_2, A_3 nie przeciął niebieskich linii wyznaczających maksymalne dopuszczalne wychylenie osi danego silnika względem jego położenia bazowego przy przemieszczaniu tej osi w kierunku dodatnim (górna linia) i w kierunku ujemnym (dolna linia),
– żaden z wykresów siły F_{A1}, F_{A2} i F_{A3} nie przeciął niebieskich linii wyznaczających maksymalną dopuszczalną siłę, z jaką operator może oddziaływać na oś danego silnika przy jej przemieszczaniu w kierunku dodatnim (górna linia) i w kierunku ujemnym (dolna linia).

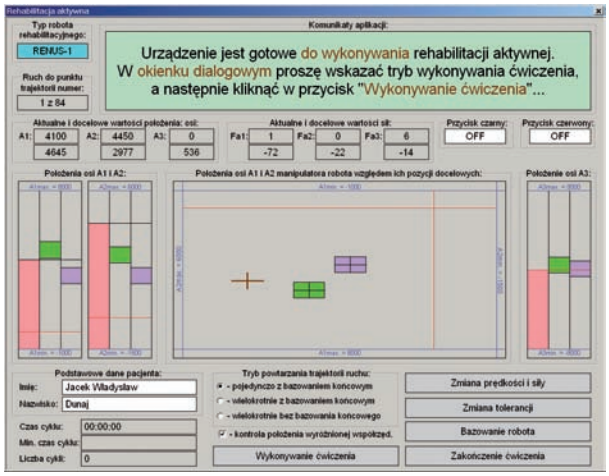
Ograniczenia te są ustalane na podstawie wartości przyporządkowanych odpowiednim zmiennym w pliku konfiguracyjnym aplikacji. Przekroczenie tylko jednego z ograniczeń spowoduje, że po zakończeniu nauki trajektoria nie zostanie zweryfikowana, a więc aplikacja Renus.exe zablokuje możliwość jej zapamiętania.

Wstrzymanie rejestracji trajektorii można wykonać w dowolnym momencie dwoma sposobami:

– przy wciśniętym czarnym sprzętowym przycisku nożnym czyli przy zwolnionych hamulcach robota – trajektoria nie jest rejestrowana, jeśli między kolejnymi odczytami położenie żadnej z trzech osi silników robota nie uległo zmianie. Informacja na ekranie monitora pozostaje wtedy „zamrożona". W przypadku robota Renus-1 manipulator może samoistnie przemieścić się w dół pod wpływem grawitacji, a więc nastąpi zmiana położenia osi co najmniej jednego z silników.
– zwolniając czarny sprzętowy przycisk nożny czyli zaciskając hamulce robota. W takim przypadku na ekranie monitora nastąpi zmiana tła wszystkich okienek z żółtego na białe. W tym przypadku operator może zakończyć naukę robota.

W drugim przypadku wznowienie rejestracji nastąpi po powtórnym wciśnięciu czarnego, sprzętowego przycisku nożnego. Przy zwolnionym przycisku aplikacja Renus.exe odblokowuje suwak umieszczony w dolnej części okna dialogowego, który umożliwia przewijanie wszystkich wykresów, o ile liczba zapamiętanych pomiarów przekracza szerokość okienek.

Rejestracja położeń i sił w wewnętrznych tablicach programu Renus.exe ma jednak na celu tylko ich prezentację w postaci wykresów i nie jest bezpośrednio związana z zapamiętywaniem trajektorii. Kolejne punkty trajektorii



Rys. 15. Rehabilitacja aktywna: oczekiwanie na zainicjowanie wykonania ćwiczenia

Fig. 15. Active rehabilitation: waiting for the beginning of the exercise

są rejestrowane co 1 sekundę przez aplikację sterownika w jego pamięci masowej, a do komputera PC automatycznie są przesyłane dopiero po zakończeniu nauki.

Aby zakończyć naukę operator powinien zwolnić czarny sprzętowy przycisk nożny, powodując zaciśnięcie hamulców, a następnie w polu dialogowym aplikacji Renus.exe do nauki robota wybrać jedną z opcji:

Anulowanie zmian: zakończenie nauki bez wprowadzania jakichkolwiek zmian,

Potwierdzenie zmian: zakończenie nauki, przesłanie trajektorii ze sterownika do komputera PC, jej weryfikacja, a po potwierdzeniu poprawności – zapamiętanie w bazie danych i pliku tekstowym.

W obu przypadkach aplikacja Renus.exe doprowadzi manipulator robota do pozycji bazowej, a następnie przywróci okno dialogowe do obsługi bazy danych trajektorii.

6. Rehabilitacja za pomocą robotów Renus

Roboty Renus umożliwiają prowadzenie dwóch rodzajów rehabilitacji:

- Rehabilitację aktywną polegającą na jednokrotnym lub wielokrotnym odtwarzaniu przez pacjenta wybranej trajektorii ruchu. Podczas wykonywania tej czynności to pacjent przemieszcza manipulator robota w przestrzeni wzdłuż wybranej trajektorii.
- Rehabilitację bierną polegającą na jednokrotnym lub wielokrotnym odtwarzaniu przez manipulator robota wybranej trajektorii ruchu. Podczas wykonywania tej czynności to robot przemieszcza kończynę pacjenta w przestrzeni wzdłuż wybranej trajektorii.

Wykonywanie obu rodzajów rehabilitacji jest więc możliwe dopiero po wybraniu i zweryfikowaniu trajektorii ruchu przy pomocy okna dialogowego do obsługi bazy danych trajektorii.

Manipulator robota ma wyróżnione położenie nazywane położeniem bazowym. Jest to położenie, dla którego położenia osi A_1, A_2, A_3 wszytkich trzech silników robota mają wartości 0. Położenie bazowe manipulatora jest niezależne od wybranej trajektorii ruchu i można je zmienić podczas serwisowej obsługi serwonapędów. Wykonywanie ćwiczenia rozpoczyna się zawsze od położenia bazowego, a po zakończeniu ćwiczenia manipulator jest także automatycznie przemieszczany do tego położenia.

Rehabilitację czynną i bierną można wykonywać w trzech trybach odtwarzania trajektorii:

- Jednokrotne odtworzenie wybranej trajektorii: Punktem startowym jest położenie bazowe, od którego pacjent przemieszcza manipulator (rehabilitacja aktywna) lub robot przemieszcza kończynę pacjenta (rehabilitacja bierna) przez kolejne punkty trajektorii aż do ostatniego zdefiniowanego punktu. Po osiągnięciu tego punktu manipulator jest automatycznie przemieszczany do położenia bazowego.
- Wielokrotne odtwarzanie wybranej trajektorii z bazowaniem końcowym: Punktem startowym jest położenie bazowe, od którego pacjent przemieszcza manipulator (rehabilitacja aktywna) lub robot przemieszcza kończynę pacjenta (rehabilitacja bierna) przez kolejne punkty trajektorii aż do ostatniego zdefiniowanego punktu. Po osiągnięciu tego punktu w obu przypadkach manipulator jest automatycznie przemieszczany do położenia bazowego, a następnie cały cykl zostaje powtórzony.
- Wielokrotne odtwarzanie wybranej trajektorii bez bazowania końcowego: Punktem startowym jest położenie bazowe, od którego pacjent przemieszcza manipulator (rehabilitacja aktywna) lub robot przemieszcza kończynę pacjenta (rehabilitacja bierna) przez kolejne punkty trajektorii aż do ostatniego zdefiniowanego punktu. Po jego osiągnięciu manipulator nie przemieszcza się automatycznie do położenia bazowego, jak w punkcie 2, ale pozwala zrealizować ruch (rehabilitacja aktywna) albo sam wymusza ruch kończyny pacjenta (rehabilitacja bierna) do pierwszego punktu trajektorii, następnie cały cykl zostaje powtórzony.

Wyboru trybu odtwarzania trajektorii można dokonać zarówno przed rozpoczęciem wykonywania ćwiczenia, jak też w dowolnym momencie jego realizacji, korzystając z okna dialogowego obsługi danego rodzaju rehabilitacji. W każdym z trzech opisanych przypadków operator stanowiska może ręcznie przerwać wykonywanie ćwiczenia.

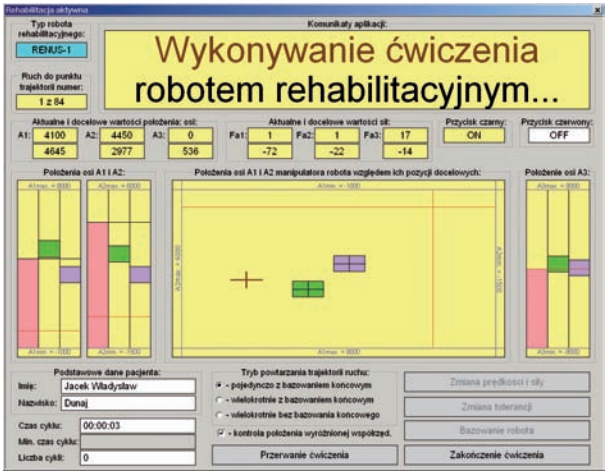
Aby było możliwe wykonywanie rehabilitacji aktywnej/biernej muszą być spełnione dwa warunki:

- komputer PC i sterownik robota powinny być sprzężone sprawnym kanałem komunikacyjnym umożliwiającym transmisję trajektorii do robota, odczyt jego rejestrów stanu oraz przesyłanie komend,
- wybrana trajektoria ruchu jest dedykowana dla aktualnie obsługiwanego typu robota i została poprawnie przesłana z komputera PC do pamięci sterownika.

Warunki te są sprawdzane bezpośrednio po wyświetleniu okna dialogowego rehabilitacji aktywnej/biernej. W trakcie weryfikacji i transmisji trajektorii, doprowadzania manipulatora do pozycji bazowej i podczas realizacji ćwiczenia rehabilitacyjnego aplikacja Renus.exe monitoruje kanał transmisyjny oraz sprawdza kody błędów odczytane z reje-

Rys. 16. Rehabilitacja aktywna: komunikat o gotowości urządzenia do obsługi wykonywania ćwiczenia

Fig. 16. Active rehabilitation: message readiness for the maitenance of exercise



Rys. 17. Rehabilitacja aktywna: okno dialogowe podczas odtwarzania trajektorii ruchu

Fig. 17. Active rehabilitation: the dialog box while playing trajectory

strów stanu robota. Każdy błąd jest błędem krytycznym, powodującym wyświetlenie komentarza z jego opisem i zakończenie realizacji ćwiczenia.

Wykonanie obu rodzajów rehabilitacji jest inicjowane przyciskiem „Wykonywanie ćwiczenia" w oknie dialogowym do obsługi rehabilitacji aktywnej (rys. 15) i biernej (rys. 20). W tym momencie zmienia się tło niektórych okienek z szarego na białe (rys. 16 i rys. 21) oraz opis przycisku na „Zakończenie ćwiczenia". Ze względów bezpieczeństwa hamulce robota pozostają zawsze zaciśnięte, jeśli zwolniony jest czarny, „sprzętowy" przycisk nożny dołączony do sterownika robota. Wciśnięcie przycisku, a więc odblokowanie hamulców jest sygnalizowane kolejną zmianą tła części okienek – tym razem na żółte (rys. 17 i rys. 22). Odtwarzanie trajektorii nie musi odbywać się aż do ostatniego punktu, ponieważ ruch manipulatora można w każdej chwili zatrzymać, zwalniając czarny przycisk, wznowienie ćwiczenia nastąpi po jego kolejnym wciśnięciu. W przypadku jednokrotnego odtwarzania trajektorii lub jej wielokrotnego odtwarzania z bazowaniem końcowym po osiągnięciu ostatniego punktu manipulator robota jest automatycznie przemieszczany do pozycji bazowej. W przypadku wielokrotnego odtwarzania trajektorii bez bazowania końcowego – aby umożliwić ręczne przemieszczenie manipulatora do jej pierwszego punktu należy zwolnić, a następnie powtórnie wcisnąć przycisk nożny, o czym informują stosowne komunikaty.

6.1. Rehabilitacja aktywna

Zadaniem pacjenta podczas wykonywania ćwiczenia jest przemieszczanie manipulatora między kolejnymi punktami trajektorii. Należy mu zatem przekazać informację o aktualnym położeniu manipulatora i o położeniach punktów docelowych. W oknie dialogowym (rys. 15, 16 i 17) informacja ta jest wyświetlana w polach opisanych komentarzami: „Aktualnie i docelowe położenia osi", „Aktualne i docelowe wartości sił". Taki sposób prezentacji, choć dobry do celów testowych, jest niewygodny dla pacjenta.

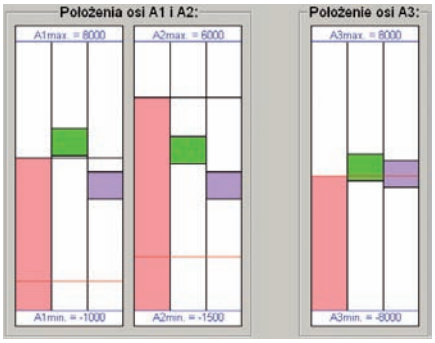
Dlatego okno dialogowe do obsługi rehabilitacji aktywnej uzupełniono o dodatkowe informacje graficzne, które w większej skali pokazano na rys. 18 i 19.

Dla zachowania przejrzystości opis został ograniczony do skrajnego lewego okienka (rys. 18). W górnej i dolnej części tego okienka wykreślono dwie niebieskie poziome linie odpowiadające:

- maksymalnej dopuszczalnej wartości położenia osi silnika nr 1 manipulatora przy jego przemieszczaniu w kierunku dodatnim (górna linia),
- maksymalnej dopuszczalnej wartości położenia osi silnika nr 1 manipulatora robota przy jego przemieszczaniu w kierunku ujemnym (dolna linia).

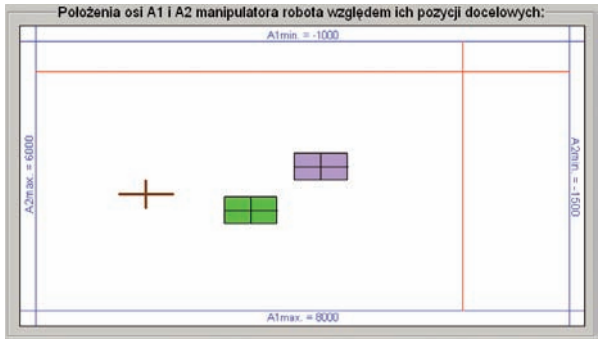
Obok każdej z tych linii są komentarze A_{1max}=... i A_{1min}=... Określają one wartości liczbowe dopuszczalnych skrajnych położeń osi silnika nr 1, ustalane niezależnie dla manipulatora Renus-1 i Renus-2 w pliku konfiguracyjnym aplikacji. Między niebieskimi liniami znajduje się pozioma czerwona linia, odpowiadająca położeniu bazowemu tej osi. W okienku są trzy prostokąty:

- Prostokąt jasnoczerwony przedstawia aktualne położenie osi silnika nr 1 względem jego pozycji bazowej, przy czym dolna krawędź prostokąta pokrywa się z niebieską linią odpowiadającą wartości A_{1min}, górna krawędź prostokąta odpowiada aktualnemu położeniu tej osi w skali od A_{1min} do A_{1max}.
- Zmiana położenia osi silnika nr 1 manipulatora względem jego pozycji bazowej powoduje zmianę wysokości jasnoczerwonego prostokąta.
- Prostokąt zielony odpowiada położeniu osi silnika nr 1 manipulatora względem jego pozycji bazowej dla najbliższego punktu trajektorii (punkt i), do którego ma zostać doprowadzony robot, przy czym położenie dolnej krawędzi prostokąta wyznaczono jako różnicę położenia osi silnika dla tego punktu trajektorii minus dopuszczalna tolerancja doprowadzenia osi tego silnika do wymaganej pozycji, a położenie górnej krawędzi prostokąta wyznaczono jako sumę położenia osi silnika dla tego punktu



Rys. 18. Fragment okna dialogowego do obsługi rehabilitacji aktywnej z okienkami do prezentacji aktualnych położeń osi silników nr 1, 2 i 3 manipulatora oraz ich położeń docelowych dla dwóch kolejnych punktów trajektorii

Fig. 18. Details of the dialog box for maitenance of the active rehabilitation with windows to present current motor axis positions 1, 2 and 3, and the target positions for two consecutive points of the trajectory



Rys. 19. Fragment okna dialogowego do obsługi rehabilitacji aktywnej do prezentacji pozycji osi silników nr 1 i 2 dla dwóch kolejnych punktów trajektorii

Fig. 19. Details of dialog box for maitenance the active rehabilitation to the presentation of positions of motor axes 1 and 2 for two consecutive trajectory points

trajektorii plus dopuszczalna tolerancja doprowadzenia osi tego silnika do wymaganej pozycji.

– Prostokąt fioletowy odpowiada położeniu osi silnika nr 1 manipulatora dla kolejnego punktu trajektorii (punkt i+1).

Zadaniem pacjenta jest takie przemieszczanie manipulatora robota, aby górna krawędź czerwonego prostokąta znalazła się wewnątrz obszaru ograniczonego dolną i górna krawędzią zielonego prostokąta. Jeśli warunek ten zostanie spełniony dla położeń osi wszystkich silników, to aplikacja uznaje, że robota doprowadzono do kolejnego punktu trajektorii, więc uaktualnia dane prezentowane w postaci zielonego i fioletowego prostokąta. Pacjent może także śledzić zmiany położeń osi silników nr 1 i 2 za pomocą informacji pokazanej w innej postaci (rys. 19).

Aplikacja wyświetla aktualne położenie osi silników nr 1 i 2 manipulatora robota oraz położenia dwóch najbliższych punktów trajektorii (w przypadku robota Renus-1 odpowiada to przemieszczaniu manipulatora w płaszczyźnie poziomej). Oprócz stałych elementów oznaczonych niebieskim i czerwonym kolorem, okienko zawiera trzy elementy o zmiennym położeniu:

Krzyż utworzony przez dwa brązowe odcinki. Środek krzyża odpowiada aktualnemu położeniu osi silników nr 1 i 2 manipulatora względem ich pozycji bazowych, a długości ramion (lewe, prawe, górne, dolne) odpowiadają dopuszczalnej tolerancji doprowadzenia osi silnika nr 1 i osi silnika nr 2 do wymaganej pozycji.

Prostokąt zielony:

– jego środek odpowiada położeniu osi silników nr 1 i 2 względem ich położeń bazowych dla najbliższego punktu trajektorii (punkt i) do którego ma zostać doprowadzony manipulator,
– długość poziomego boku prostokąta odpowiada podwojonej, dopuszczalnej tolerancji doprowadzenia osi silnika nr 1 do wymaganej pozycji,
– długość pionowego boku prostokąta odpowiada podwojonej, dopuszczalnej tolerancji doprowadzenia osi silnika nr 2 do wymaganej pozycji.

Prostokąt fioletowy odpowiada położeniu osi silników nr 1 i 2 dla kolejnego punktu trajektorii (punkt i+1).

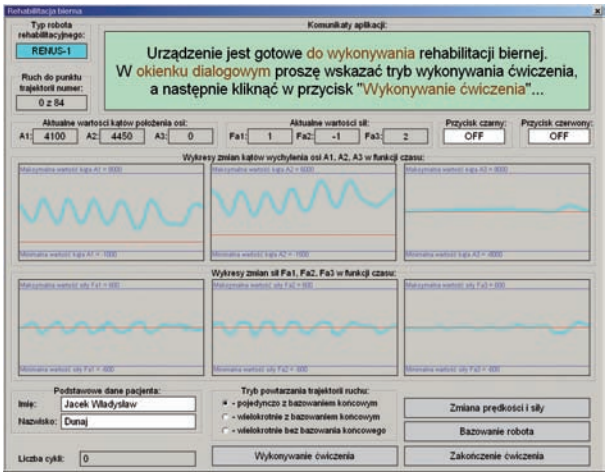
Zadaniem pacjenta jest takie przemieszczanie manipulatora robota, aby środek brązowego krzyża znalazł się wewnątrz zielonego prostokąta. Praktycznie podczas wykonywania ćwiczenia pacjent powinien obserwować okienko (rys. 19) oraz wykres słupkowy dotyczący położenia osi silnika nr 3.

W praktyce dokładne doprowadzenie manipulatora do określonego punktu w przestrzeni jest trudne, szczególnie dla osoby z upośledzoną kończyną. Dlatego podczas wykonywania ćwiczenia zadaniem pacjenta nie jest dokładne przemieszczenie manipulatora do danego punktu trajektorii, ale doprowadzenie go w sąsiedztwo tego punktu. O tym, jak blisko punktu powinien być doprowadzony manipulator, aby potwierdzić wykonanie próby decydują wartości tolerancji dojścia do punktu, ustawiane indywidualnie dla każdego silnika. Zmiana tolerancji, wykonywana przy pomocy dodatkowego okna dialogowego aplikacji Renus.exe, powoduje zmianę wymiarów zielonych i fioletowych prostokątów (rys. 15–19).

Obecnie jedynym kryterium oceny postępów pacjenta jest pomiar czasu przemieszczenia manipulatora wzdłuż całej trajektorii i porównania go z czasami poprzednich prób. Wartość ta nie jest zapisywana w bazie danych.

6.2. Rehabilitacja bierna

Rehabilitacja bierna wspomagana robotem Renus jest mniej stresująca dla pacjenta, ponieważ nie wiąże się z koniecznością śledzenia licznych komunikatów i wykresów wyświetlanych na ekranie komputera PC. Jedynym zadaniem pacjenta jest biernie poddawanie jego kończyny przemieszczeniu wymuszonemu przez manipulator, a w dalszej fazie ćwiczeń – przeciwstawianie się temu wymuszeniu. Widoki okien dialogowych aplikacji Renus.exe w różnych fazach realizacji tej rehabilitacji pokazano na rys. 20, 21 i 22. W centralnej części okien dialogowych znajduje się sześć pól, w których prezentowane są wykresy zmian położeń A_1,

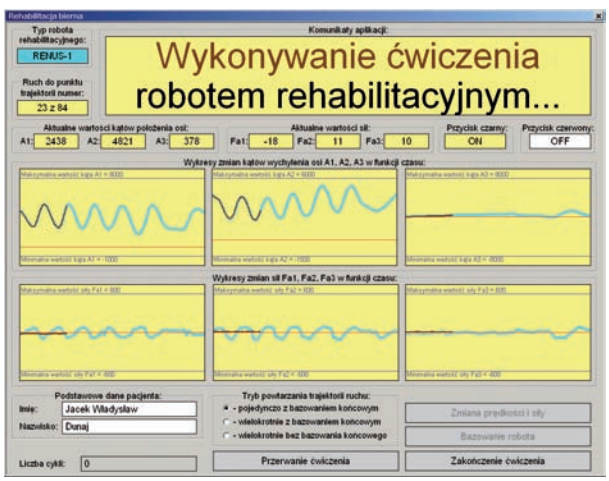


Rys. 20. Rehabilitacja bierna: Oczekiwanie na zainicjowanie wykonania ćwiczenia

Fig. 20. Passive rehabilitation: Waiting for the beginning of exercise

Rys. 21. Rehabilitacja bierna: Komunikat o gotowości urządzenia do obsługi wykonywania ćwiczenia

Fig. 21. Passive rehabilitation: Message readiness for the maitenance of exercise



Rys. 22. Rehabilitacja bierna: Okno dialogowe podczas odtwarzania trajektorii ruchu

Fig. 22 Passive rehabilitation: The dialog box while playing trajectory

A_2 i A_3 osi trzech silników robota (turkusowe linie) oraz wykresy zmian sił F_{A1}, F_{A2} i F_{A3} oddziaływania osi silników na operatora w funkcji czasu dla całej realizowanej trajektori ruchu. Aktualne przebiegi zmian tych położeń i sił przedstawiają wykresy wyświetlane jako cienkie czarne linie (rys. 22).

7. Wnioski

Roboty rehabilitacyjne stają się coraz bardziej znaczącymi urządzeniami w usprawnianiu pacjentów po udarach mózgu i ze schorzeniami ortopedycznymi. Systematyczna i odpowiednio prowadzona rehabilitacja ruchowa jest skutecznym sposobem poprawy sprawności ruchowej pacjentów dotkniętych tymi schorzeniami. Rehabilitacja ta wymaga od pacjenta mozolnej pracy polegającej na wielokrotnym powtarzania zadanych przez fizjotrapeutę ćwiczeń z użyciem odpowiedniego sprzętu medycznego. W wielu przypadkach takie ćwiczenia są wykonywane z bezpośrednim udziałem fizjoterapeuty i często przy znacznym wysiłku fizycznym z jego strony. Szybkie tempo życia i stres powodują, że rośnie liczba ludzi doznających udarów mózgu. Ich powrót do czynnego życia zawodowego i społecznego staje się możliwy w wyniku leczenia oraz prawidłowej rehabilitacji. Stąd też od wielu lat intensywnie poszukuje się nowych, skutecznych i oferowanych po rozsądnych cenach urządzeń, które mogą wspomagać fizjoterapetów w ich pracy z pacjentem. Przeprowadzone badania kliniczne rehabilitacji ruchowej pacjentów z użyciem robotów wspomagających bezspornie wykazały wartość terapeutyczną tych urządzeń i nowych metod rehabilitacji. W wielu przypadkach rehabilitacja z użyciem takich robotów jest bardziej skuteczna niż w przypadku klasycznych metod [5]. Stąd też prace nad powstaniem nowych zrobotyzowanych urządzeń rehabilitacyjnych są obecnie prowadzone w wielu ośrodkach badawczych na świecie i w Polsce. W Przemysłowym Instytucie Automatyki i Pomiarów PIAP prace te rozpoczęto w 2005 r. Koncentrują się one na konstrukcjach mechanicznych, sterowaniu i oprogramowaniu.

Pomysł zastosowania „przemysłowych" elementów sterujących w konstrukcji tak odległej od przemysłu jak robot rehabilitacyjny okazał się trafny. Zastosowanie gotowych elementów firmy Mitsubishi Electric pozwoliło uniknąć większości problemów związanych z opracowaniem i uruchomieniem układu sterowania i jego oprogramowania systemowego. W warunkach PIAP byłoby to oczywiście możliwe, ale wymagałoby dodatkowych środków i nakładu pracy. To, że Instytut dysponuje działającymi modelami urządzeń jest w dużej mierze konsekwencją tej decyzji. Nawet taka z pozoru błaha sprawa jak opracowanie bibliotek do niezawodnej komunikacji jednostka centralna – aplikacja komputera PC wymagają szeregu testów i sprawdzenia, czy w szczególnych warunkach komunikacja ta nie zawodzi i negatywnie nie wpływa na inne funkcje urządzenia. Firma Mitsubishi to wszystko dostarczyła, co więcej świadczy również usługi serwisowe na zastosowane elementy sterowania.

Oba roboty rehabilitacyjne Renus-1 i Renus-2 są obecnie w pełni funkcjonalnymi, działającymi modelami. Były

one wielokrotnie prezentowane osobom ze środowiska medycznego, wzbudzając zainteresowanie, w tym dotyczące terminu rozpoczęcia ich produkcji i ich potencjalnej ceny. Dalsze prace w PIAP będą koncentrować się na konstrukcji robotów rehabilitacyjnych, ich sterowania i oprogramowania. Roboty te powinny być alternatywą do bardzo drogich urządzeń podobnego typu oferowanych w Polsce przez czołowe firmy europejskie i światowe. Tematyka robotów rehabilitacyjnych stała się na tyle popularna, że autorzy opracowania obu robotów Renus kilkukrotnie w ciągu roku otrzymują prośby o materiały na temat ich budowy i oprogramowania.

Podziękowanie

Publikacja powstała jako wynik projektu *RoboReha – Robotics in Rehabilitation*, LdV – TOI no. 13310 0530, realizowanego przy wsparciu finansowym Komisji Europejskiej w ramach programu *Uczenie się przez całe życie*. Publikacja odzwierciedla jedynie stanowisko autorów. Komisja Europejska ani Narodowa Agencja nie ponoszą odpowiedzialności za umieszczoną w niej zawartość merytoryczną ani za sposób wykorzystania zawartych w niej informacji.

Bibliografia

1. Dunaj J., *Opis działania, konfiguracji i obsługi aplikacji „Renus" do sterowania robotami rehabilitacyjnymi Renus-1 i Renus-2*, Materiały Przemysłowego Instytutu Automatyki i Pomiarów PIAP, czerwiec 2014.
2. Dunaj J., Pilat Z., Klimasara W.J., *System komunikacji człowiek–robot w urządzeniach rehabilitacyjnych*, [w:] Materiały XIII Krajowej Konferencji Robotyki, Kudowa Zdrój, 2014.
3. Klimasara W.J., *Roboty w rehabilitacji osób po udarze mózgu*, AUTOMATION 2003. Konferencja Naukowo-Techniczna Automatyzacja – Nowości i Perspektywy. Warszawa, 2–4 kwietnia 2003, Zbiór referatów, 417–424.
4. Klimasara W.J., *Raporty i sprawozdania końcowe z realizacji zadań badawczych projektu badawczego zamawianego nr PW-004/ITE/02/2006.*
5. Krebs H.I., Palazzolo J.J., Dipietro L., Ferraro M., Krol J., Rannekleiv K., Volpe B.T., Hogan N., *Rehabilitation Robotics: Performance-Based Progressive Robot-Assisted Therapy*, „Auton. Robots", 15(1): 7–20, 2003.
6. Laidler P., *Rehabilitacja po udarze mózgu. Zasady i strategia*, PZWL, Warszawa 2006.
7. Mitsubishi Electric Corporation 2002 – MX Component Programming Manual.
8. Mitsubishi Electric Corporation 2002 – MX Component Operating Manual.
9. Microsoft Visual C++ 6.0 MFC Library Reference, Microsoft Press 1998.
10. Bates J., Tompkins T., *Poznaj Visual C++*, Wydawnictwo MIKOM, 1999.
11. Petzold Ch., *Programowanie Windows*, Wydawnictwo RM, Warszawa 1999.

Hardware and software solutions in the control of rehabilitation robots Renus

Abstract: This paper descibes how the control and software solutions in the Renus 1 and Renus 2 robots for support patients after strokes and orthopedic diseases were implemented. Control of both robots was performed on the basis of commercial elements of Mitsubishi Electric used in industrial automation. Their use has allowed to build a fully functional models of the two robots in a relatively short time. The paper presents the structure and control software solutions used in developing software both robots. Describes the user interface human–robot for both the patient and the operator – a physiotherapist. Interface description is illustrated by the prints screens. Both devices Renus have been developed in the Industrial Research Institute for Automation and Measurements.

Keywords: neurological rehabilitation, rehabilitation robot, PLC, a PC application, database, ODBC drivers

Artykuł recenzowany, nadesłany 10.10.2014 r., przyjęty do druku 07.11.2014 r.

mgr inż. Jacek Dunaj

W 1980 r. ukończył studia na Wydziale Elektrycznym Politechniki Warszawskiej, od 1985 r. jest zatrudniony w Przemysłowym Instytucie Automatyki i Pomiarów PIAP. Specjalizuje się w programowaniu mikroprocesorów, kontrolerów, sterowników i robotów przemysłowych, systemów wizyjnych a także komputerów PC (asembler, C/C++) w różnych systemach operacyjnych. Współautor oprogramowania dla kilku urządzeń opracowanych w PIAP, w tym robotów Renus, a także wielu wdrożeń przemysłowych, w szczególności wymagających współpracy ze sobą kilku różnych urządzeń automatyki i wykorzystania oprogramowania biurowego (baz danych, arkuszy kalkulacyjnych).

e-mail: jdunaj@piap.pl

mgr inż. Wojciech J. Klimasara

Absolwent Politechniki Warszawskiej. Studia z wyróżnieniem ukończył na Wydziale Mechaniki Precyzyjnej w 1974 r. Od 1978 r. pracuje w Przemysłowym Instytucie Automatyki i Pomiarów PIAP specjalizując się w problematyce robotów. Kierował i brał udział w wielu krajowych i zagranicznych projektach badawczych. Jest współtwórcą pierwszego polskiego robota interwencyjno-inspekcyjnego Inspector. Tematyka zainteresowań: projektowanie, badania oraz zastosowania robotów i manipulatorów do pracy w warunkach trudnych i niebezpiecznych, interfejsy człowiek–robot, robotyka rehabilitacyjna. Jest autorem oraz współautorem kilkunastu patentów oraz kilkudziesięciu publikacji: referatów i artykułów w czasopismach krajowych i zagranicznych.

e-mail: wklimasara@piap.pl