www.pl2022.pollub.pl
e-mail: pl2022@pollub.pl

Zintegrowany Program Rozwoju Politechniki Lubelskiej

Biuro Projektu:
ul.Nadbystrzycka 38H
20 -618 Lublin

Komputerowe wspomaganie projektowania stanowisk zrobotyzowanych

Workbook

Radosław Cechowicz
Jarosław Zubrzycki

Lublin, 2021

PROGRAM WIEDZA EDUKACJA ROZWÓJ

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Unia Europejska Europejski Fundusz Społeczny

Spis treści

1 Wprowadzenie

Współczesne przedsiębiorstwo produkcyjne aby zachować swoją pozycję na rynku jest zobligowane do stawiania na najnowocześniejsze zdobycze zarówno techniki jak i zarządzania procesem produkcyjnym. Nieodzownym środkiem do utrzymania konkurencyjności na rynku pracy jest utrzymanie wysokich standardów jakości wykonania produktów jak też wysoka wydajność i krótkie terminy realizacji zamówienia.

Aby sprostać tym wyzwaniom koniecznością staje się stawianie na robotyzację procesów wytwarzania. Robotyzacja produkcji to jedna ze składowych organizacji przemysłu – Industry 4.0.

Wprowadzanie robotów do procesu produkcyjnego i technologicznego wymaga szeregu zabiegów technicznych i organizacyjnych mających na celu zastąpienie części czynności wykonywanych przez człowieka pracami wykonywanymi przez roboty przemysłowe. Wprowadzenie do produkcji robotów wymaga spełnienia szeregu wymogów narzucanych przez otoczenie. Zastosowanie robotów pociąga za sobą m.in. zapewnienie bezpiecznej pracy dla współpracującego z robotem człowieka. Klasyk robotyzacji powiedział: „Robot nie może zranić człowieka ani poprzez zaniechanie działania dopuścić, aby człowiek doznał obrażeń” – Isaak Asimov 1 prawo robotyki. Jedno z praw robotyki Asimova, które sformułował w swoim opowiadaniu pt. Runaround (Zabawa w berka 1942).

Na szczególną uwagę zasługuje jedna z jego wypowiedzi dla amerykańskiej prasy: „Zmiana, ciągła zmiana, nieuchronna zmiana. Jest dominującym czynnikiem w dzisiejszym społeczeństwie. Żadna rozsądna decyzja nie może być już podjęta bez uwzględnienia nie tylko świata, jaki jest, ale także bez uwzględnienia świata, jaki będzie - a to oczywiście oznacza, że musi istnieć dokładne postrzeganie świata, jaki będzie. To z kolei oznacza, że nasi mężowie stanu, nasi biznesmeni, nasi everymani muszą przyjąć sposób myślenia rodem z fantastyki naukowej.” Można więc stwierdzić, że każdy inżynier zajmujący się projektowaniem stanowisk zrobotyzowanych powinien być zaznajomiony z aktualnymi wydarzeniami i otwarty na innowacyjne sposoby myślenia wywołane zmianą okoliczności zewnętrznych.

Bezpieczeństwo pracy z robotami jest naczelnym tematem dużej liczby opowiadań, powieści, słuchowisk radiowych i filmów science-fiction, które zachęcają do patrzenia w przyszłość i myślenia o niej poprzez pokazywanie wyzwań, jakie stoją przed ludzkością intensywnie inwestującą w rozwijanie technik i technologii robotycznych. Celem wielu takich utworów tworzonych przez twórców z kręgu sci-fi jest i było pokazanie przyszłości pod wpływem nowych technologii. Bo to, co dzisiaj jest tylko fikcją literacką jutro staje się rzeczywistością. Rzeczywistością, która źle zorganizowana i niewłaściwie zabezpieczona może poważnie namieszać i zaszkodzić człowiekowi i wszystkiemu co, stworzył przez ostatnie tysiąclecia rozwoju ludzkości.

2 Prawa robotyki Asimowa

Manipulacyjne roboty przemysłowe to wielofunkcyjne urządzenia mechaniczne, które można zaprogramować na potrzeby wykonywania różnych zadań. System robotów manipulacyjnych obejmuje nie tylko roboty przemysłowe, ale także różnego rodzaju urządzenia czy czujniki wymagane do wykonywania zadań robota. Roboty stosuje się zazwyczaj do wykonywania niebezpiecznych, często powtarzalnych i trudnych zadań. Pełnią wiele funkcji, m.in. są wykorzystywane do: transportu, montażu, malowania lub spawania. W większości są skonfigurowane do wykonywania operacji techniką teach-and-repeat. Problematyka bezpieczeństwa pracy z systemami robotycznymi oraz standardami bezpieczeństwa jest szeroko omawiana w cenionych czasopismach o ogólnoświatowym zasięgu (np. Robotics Safety Science, Journal of Robotic Systems i inne). Aby stworzyć bezpieczne stanowisko pracy z robotem należy wykonać dwie podstawowe czynności:

- Sporządzić ocenę ryzyka panującego na zrobotyzowanym stanowisku,
- Sporządzić listę czynności zabezpieczeń ograniczających wystąpienie ryzyka.

Robotyzacja i roboty były wiodącymi tematami prac Asimova, które z biegiem lat tylko zyskały na aktualności. Początkowo tylko fikcyjne pomysły wpłynęły na rozwój technologii przyszłości. Asimov w szeregu swoich prac formułował prognozy i kreował świat robotyki przyszłości. Wyobrażał sobie, jak mogą rozwijać się i działać technologie i jaka w tym wszystkim będzie rola i miejsce człowieka, jak będą współdziałać ze sobą człowiek i maszyna wyposażona w inteligencję. Bazując na swoich przemyśleniach i wizjach w roku 1942 w opowiadaniu „Zabawa w berka" jako wizjoner, który wyprzedzał swoje czasy, jako pisarz science fiction zatroskany o przyszłość ludzkości przewidział konieczność wprowadzenia swoiście rozumianej moralności do robotów. Jest to najwcześniejsza forma czegoś, co dziś nazywamy robotyką. Celem tych zabiegów jest zapewnienie bezpieczeństwa ludziom w kontaktach z robotami, ale także ma stanowić ochronę człowieka przez robota. Postulaty Asimova związane z bezpieczeństwem prazy z robotami przeszły do historii jako „Prawa robotyki Asimova" i są to:

1. Robot nie może zranić człowieka ani przez zaniechanie działania pozwolić, aby człowiek doznał obrażeń.
2. Robot musi być posłuszny rozkazom człowieka, chyba że stoją one w sprzeczności z Pierwszym Prawem.
3. Robot musi chronić samego siebie, o ile tylko nie stoi to w sprzeczności z Pierwszym lub Drugim Prawem.

Mimo wszystko, nie był do końca i w pełni usatysfakcjonowany sformułowanymi prawami. Wiele lat później bo w roku 1985, opublikował swoją nowelę s-f pt. Robots and Empire (Roboty i Imperium), w której zdefiniował nadrzędne prawo robotyki tzw. „prawo zerowe”:

Robot nie może skrzywdzić ludzkości, lub poprzez zaniechanie działania doprowadzić do uszczerbku dla ludzkości.

Sama składnia treści trzech pierwotnych praw robotów Asimova podkreśla referencyjność; ich zależność względem siebie. O co tu chodzi? Przed około 1940 rokiem prawie każda historia ze zbioru tzw. fikcji spekulatywnej (od ang. speculative fiction) – czyli science-fiction, fantasy, historia alternatywna i wszystko, co podobne, obok i pomiędzy – z udziałem robotów podążała za modelem Frankensteina. Robot musiał więc nieustannie otrzymywać od człowieka instrukcje, co ma robić, a przy braku takiej albo jakiejkolwiek innej kontroli właściwie wpadał w szał. Młody Isaac Asimov miał tego dość i postanowił pisać historie o sympatycznych robotach z zaprogramowanymi zabezpieczeniami, które nie pozwalałyby im na wzniecanie awantur. Rozmowa z jego redaktorem, Johnem W. Campbellem, pomogła mu ułożyć te zabezpieczenia w prawa.

Rzecz w tym, że prawa Asimova obejmują większość oczywistych sytuacji, ale daleko im do bezbłędności. Pisarz pokonał daleką drogę i napisał jeszcze wiele o tym, jak prawa kolidują ze sobą i generują nieprzewidywalne zachowania. W wielu z nich powracającą postacią była „robopsycholog” Susan Calvin, która, nie całkiem przypadkowo, była genialnym logikiem nienawidzącym ludzi. Cóż, zawód adekwatny nie tylko do potrzeb nowego świata, ale i do profilu zawodowego kobiety go wykonującej.

Można zatem śmiało, że prawa Asimova są naturalną odpowiedzią w obliczu koncepcji, gdzie roboty staną się zjawiskiem tak pospolitym jak telefony komórkowe, które jeszcze 20 lat temu były nowinką techniczną z pogranicza fikcji techniczno-naukowej. I w tym aspekcie chodzi o to, aby tak programować wewnętrznie roboty żeby nie miały one ochoty wyrządzać krzywdy człowiekowi (bunt maszyn – częsty motyw różnych publikacji i filmów s-f).

Z przykrością należy stwierdzić, że prawa Asimova zostały „nieco zapomniane”, kiedy weźmiemy pod uwagę kierowane przez człowieka roboty bojowe, jakimi są drony bojowe zaprojektowanie głównie z myślą o zabijaniu. Jest to całkowicie w opozycji do praw Asimova. Paradoksalnie rzecz biorąc, jeżeli sterowany przez człowieka robot ma ratować życie sterującego i innych ludzi zabijając innych ludzi, którzy ich atakują, to można powiedzieć, że przestrzega pierwszego prawa i jednocześnie je łamie. Ponadto jeżeli dron jest sterowany przez człowieka, to można argumentować, że błąd leży po stronie człowieka, a nie drona – na przykład w przypadku utraty życia w sytuacjach bojowych. W wojskach wyposażonych w drony znacznie spadnie ogólna liczba ofiar wśród ludzi. Ale czy na pewno tak będzie, gdy uwzględnimy prawa robotów Asimova?

Podczas gdy prawa Asimova przez dekady przyświecały projektantom robotów, to najprawdopodobniej nadszedł czas na ponowną ocenę ich skuteczności i otwarcie dyskusji nad nowym zbiorem praw, który sprawdzi się w świetle wzbudzających podziw osiągnięć w robotyce i technologii sztucznej inteligencji.

Współcześnie żyjący twórcy bardzo chętnie sięgają po tematykę robotów i ich interakcje i współdziałanie z człowiekiem. Szczególnie jest to widoczne u twórców z Dalekiego Wschodu (Japonia, Korea Południowa). Tak popularne w kręgach tamtejszej kultury anime i manga pełne są niemal apokaliptycznej wizji rozwoju robotyki. Zrealizowany w wersji kinowej Ghost in the Shell ze Scarlet Johanson w roli tytułowej (pierwowzorem była manga o tym samym tytule) przedstawia świat gynoid – robotów przypominających wyglądem dojrzałe kobiety, które łamią prawo zwane Kodeks Moralny #3 poprzez celowe tworzenie usterek we własnym oprogramowaniu.

W tym duchu Astro Boy z 2009 roku. Osamu Tezuki prawdopodobnie opracował swoje zasady niezależnie od Asimova i jest ich więcej. Oprócz zwykłego „Nie krzywdź ludzi" sformułował inne prawa, takie jak prawo zabraniające robotom podróżowania za granicę (chyba że zostanie udzielone pozwolenie), zakaz zachowywania się jak dzieci dorosłym robotom i zakaz przeprogramowywania przypisanej robotowi płci. Jednak już pierwsze prawo mówi, że roboty mają uszczęśliwiać ludzi. Tezuka podobno nie lubił praw Asimova z powodu implikacji, że czujący, sztucznie inteligentny robot nie może być uważany za osobę, i wymyślił swoje własne prawa robotyki:

1. Roboty muszą służyć ludzkości.
2. Roboty nie mogą zabijać ani krzywdzić ludzi.
3. Robot musi nazywać swojego ludzkiego stwórcę „ojcem".
4. Robot może stworzyć wszystko z wyjątkiem pieniędzy.
5. Robot nie może wyjeżdżać za granicę bez pozwolenia.
6. Roboty płci męskiej i żeńskiej nie mogą zmieniać swojej płci.
7. Robot nie może zmieniać swojej twarzy, aby stać się innym robotem.
8. Robot, który został stworzony jako dorosły, nie może stać się dzieckiem.
9. Robot nie może ponownie złożyć robota, który został zdemontowany przez człowieka.
10. Roboty nie mogą niszczyć ludzkich domów ani narzędzi.

No cóż, powyższe prawa zapewne są na wyrost dla poczciwego 6-osiowego ramienia robotycznego wykonującego powtarzające się w nieskończoność te same czynności z posuniętą do absurdu dokładnością i powtarzalnością. Ale są jak najbardziej na miejscu w przypadku jednostek wyposażonych w sztuczną inteligencję i rozwijających procesy myślowe i poznawcze, adaptujące

się do zmieniającego się środowiska, do odmiennych charakterów swoich twórców i użytkowników, którzy nie zawsze funkcjonują zgodnie z prawami Asimova.

Isaak Asimov żył wystarczająco długo, żeby na własne oczy zobaczyć jak jego prawa są inkludowane do systemów robotycznych, dało mu to dużą satysfakcję. Jednakże, aby wprowadzić w życie jego prawa konieczne jest zaaplikowanie całej grupy czynników, które pozwolą robotowi ich przestrzegać. Czynniki te to przede wszystkim:

- rozpoznawanie istoty ludzkiej (np. test Turinga[1]), jako różnej od wszystkiego innego,
- zdolność rozumowania pozwalająca rozróżnić czym jest krzywda i ocenić jej poziomy,
- zdolność przetwarzania języka naturalnego w celu określenia „rozkazów" w odróżnieniu od „sugestii",
- takie programowanie robotów, aby stały się najlepszymi przyjaciółmi człowieka.

3 Projektowanie bezpiecznego zrobotyzowanego stanowiska pracy

Linie produkcyjne na, których pracują roboty nikogo nie dziwią. Jednym z ważniejszych warunków miejsca pracy człowieka jest zapewnienie mu bezpiecznej pracy. Jest to warunek konieczny, który musi być spełniony szczególnie przez integratorów projektujących zrobotyzowane stanowiska pracy. Głównym celem projektantów takich stanowiska jest zapewnienie najwyższego stopnia bezpieczeństwa dla otoczenia robota podczas jego pracy.

[1] Test Turinga – sposób określania zdolności maszyny do posługiwania się językiem naturalnym i pośrednio mającym dowodzić opanowania przez nią umiejętności myślenia w sposób podobny do ludzkiego. W 1950 roku Alan Turing zaproponował ten test w ramach badań nad stworzeniem sztucznej inteligencji – zamianę pełnego emocji i w jego pojęciu bezsensownego pytania *Czy maszyny myślą?* na pytanie lepiej zdefiniowane. Test wygląda następująco: sędzia – człowiek – prowadzi rozmowę w języku naturalnym z pozostałymi stronami. Jeśli sędzia nie jest w stanie wiarygodnie określić, czy któraś ze stron jest maszyną czy człowiekiem, wtedy mówi się, że maszyna przeszła test. Zakłada się, że zarówno człowiek, jak i maszyna próbują przejść test zachowując się w sposób możliwie zbliżony do ludzkiego. Test pochodzi od zabaw polegających na zgadywaniu płci osoby znajdującej się w innym pokoju przy pomocy serii pytań i odpowiedzi pisanych na kartkach papieru. W pierwotnym pomyśle Turinga człowiek musiał udawać przeciwną płeć, a test był ograniczony do pięciominutowej rozmowy. Dziś nie uważa się tych cech za podstawowe i zasadniczo nie umieszcza w specyfikacji testu Turinga. Turing oczekiwał, że maszyny w końcu będą w stanie przejść ten test. Ocenił, że około roku 2000 maszyny z pamięcią o pojemności 10^9 bitów (około 119 MB) będą w stanie oszukać 30% sędziów w czasie pięciominutowego testu. Przewidywał również, że ludzie przestaną uważać frazę „myśląca maszyna" za wewnętrznie sprzeczną. Oceniał, że uczenie maszynowe nabierze dużego znaczenia w budowaniu wydajnych maszyn. To twierdzenie jest przez dzisiejszych badaczy sztucznej inteligencji oceniane jako zasadne.

W klasycznym ujęciu stanowisk zrobotyzowanych strefa pracy robota jest oddzielona od strefy pracy operatora szeregiem zabezpieczeń technicznych (np. wszelkiego rodzaju bariery optyczne, osłony ogrodzenia itp.), które wydzielają przestrzeń roboczą robota. Nawet panel programowania robota jest umieszczony z dala od strefy pracy robota, aby nie doprowadzić do bezpośredniego kontaktu operatora-programisty robotem.

Nowoczesne roboty, szczególnie kooperacyjne (współpracujące) nie pracują w zamkniętych, odizolowanych strefach. Często w procesie produkcyjnym mamy do czynienia z zamkniętymi zrobotyzowanymi stanowiskami, ale różne specyficzne części procesu produkcyjnego wymagają, aby strefy realizacji tych części procesu produkcyjnego nie były wygrodzone i niedostępne dla pracowników. W takich przypadkach należy zastosować wszystkie dostępne rozwiązania techniczne umożliwiające robotowi wykrycie obecności człowieka w obszarze pracy maszyny.

Odpowiedzialnymi za wykrywanie obecności człowieka w strefach otwartych mają zapewnić specjalne maty, listwy mocowane na podłodze wykrywające nacisk pochodzący od człowieka, różnego rodzaju skanery, czujniki ruchu, czujniki podczerwone temperatury itp. Dodatkowo, stosowane są różnego rodzaju systemy ostrzegawcze takie jak migające lampy ostrzegawcze czy syreny dźwiękowe.

Nawet przy projektowaniu najprostszego stanowiska zrobotyzowanego ważne jest przestrzeganie kilku niezwykle istotnych zasad bezpieczeństwa. Projektanci systemów zrobotyzowanych są zobowiązani, by w procesie projektowania uwzględniać aktualne wymagania prawne w tym zakresie, najlepiej w oparciu o aktualnie obowiązujące normy bezpieczeństwa.

Obecnie – równolegle do tradycyjnej robotyki przemysłowej – rozwija się nowy kierunek tzw. kobotów, czyli robotów współpracujących. Maszyny nieukończone, jakimi są tego typu manipulatory z założenia dopuszcza się do kontaktu z użytkownikiem. Wpisuje się to w coraz bardziej aktualny kierunek, gdzie nacisk kładzie się, ze względu na coraz droższą powierzchnię, by zrobotyzowane stanowiska projektowane były bez tradycyjnych, zajmujących dużo miejsca wygrodzeń.

Wszędzie tam, gdzie roboty pracują obok człowieka, kooperują, lecz nie występuje między nimi żadna interakcja można wirtualnie rozdzielić przestrzenie pracy człowieka i robota. W tym aspekcie, w największym stopniu, to na integratorach spoczywa ogromna odpowiedzialność, gdzie z jednej strony muszą sprostać wymaganiom ze strony przedsiębiorcy, aby maksymalnie wykorzystać robota, z drugiej strony, najważniejsze jest zapewnienie najwyższego poziomu bezpieczeństwa – bezpiecznych pozycji, zatrzymania i odpowiednio wolniejszej pracy robota w trakcie obecności operatora w bliskim sąsiedztwie robota.

Współpraca z robotem musi być bezpieczna przez zastosowanie jednego z rozwiązań. Zupełnie oddzielony od człowieka, kooperujący z nim we wspólnej przestrzeni, ale z adaptacją dynamiki do warunków pracy, albo współpracujący z człowiekiem, będący już do takiej pracy zaprojektowanym. Warto także podkreślić, iż niezależnie od zastosowanych rozwiązań technicznych, niezwykle istotnym jest właściwe przeszkolenie personelu w zakresie bezpiecznej i efektywnej pracy na stanowisku zrobotyzowanym, i tu nic nie zastąpi zdrowego rozsądku i troski o własne bezpieczeństwo obsługi podczas wykonywania pracy.

Robotyka przemysłowa jest to dziedzina robotyki zajmująca się zastosowaniem robotów i manipulatorów przemysłowych w celu robotyzacji procesów produkcyjnych (ISO 8373:2012). Rozróżnia się trzy klasy robotów:

- biotechniczne (z ręcznym sterowaniem, kopiujące z jednostronnym działaniem i dwustronnym działaniem, półautomatyczne),
- automatyczne (programowalne, adaptacyjne, intelektualne z elementami inteligencji maszynowej, roboty manipulacyjne),
- interakcyjne (zautomatyzowane, z nadrzędnym sterowaniem, dialogowe).

Stacjonarne roboty o szeregowym układzie kinematycznym można podzielić ze względu na ich strukturę kinematyczną (układ zespołów ruchu) definiowaną przez tzw. naturalny dla danej struktury układ osi współrzędnych i formę przestrzeni roboczej. Są to:

- roboty w układzie kartezjańskim, (prostokątny układ osi współrzędnych o trzech liniowych zespołach ruchu regionalnego oraz prostopadłościennych przestrzeniach ruchu),
- roboty w układzie cylindrycznym (jeden obrotowy i dwa liniowe zespoły ruchu regionalnego, walcowy układ osi współrzędnych i cylindryczne przestrzenie ruchu),
- roboty SCARA (ang. selectively compliant assembly robot arm), zaprojektowane do zadań montażowych (trzy osie równoległe, dwie o ruchu obrotowym i jedna o postępowym),
- roboty PUMA (ang. programmable universal manipulator for assembly), również przeznaczone do zadań montażowych (struktura kinematyczna jak u robota przegubowego; różnią się od niego jedynie wyglądem i możliwościami zastosowania,
- roboty o strukturze sferycznej (jeden liniowy oraz dwa obrotowe zespoły ruchu regionalnego),
- roboty o strukturze przegubowej, zwane też manipulatorami obrotowymi (wszystkie obrotowe osie zespołów ruchu regionalnego),
- roboty wielokorbowe.

Jak łatwo zauważyć, przestrzeń robocza robota przemysłowego determinuje określenie strefy niebezpiecznej dla otoczenia, czyli jest to strefa wokół maszyny, w której występuje realne zagrożenie bezpieczeństwa lub zdrowia osób znajdujących się w niej (nawet częściowo).

Aby zapewnić bezpieczną pracę człowieka w otoczeniu robota realizuje się różna programy bezpieczeństwa, które mają na celu ograniczenie ryzyka obrażeń ciała lub mogłyby mieć negatywny wpływ na funkcjonowanie przedsiębiorstwa. W tym aspekcie istnieje szereg typów programów zapewniających bezpieczną pracę w przemyśle produkcyjnym wykorzystującym roboty.

- Bezpieczeństwo i higiena pracy - programy szkoleniowe i edukacyjne dla pracowników z zakresu unikania obrażeń ciała,
- Bezpieczeństwo produktów lub ostrzeżenia bezpieczeństwa w zakresie prawidłowego użytkowania, konserwacji oraz napraw maszyn i urządzeń,
- Bezpieczeństwo maszyn i układy zabezpieczeń, wraz z zabezpieczeniami fizycznymi, sterowaniem oraz procedurami, takimi jak blokady zabezpieczające i oznakowanie ostrzegawcze (LOTO),
- Bezpieczeństwo środowiskowe – programy unikania zanieczyszczenia powietrza lub ziemi,
- Bezpieczeństwo mienia i urządzeń oraz ochrona inwestycji kapitałowych, np. instalacja automatycznego układu zraszaczy.

W skład programu bezpieczeństwa maszyn wchodzą: analiza ryzyka, środki ograniczające ryzyko oraz szkolenia. Etap analizy ryzyka obejmuje określenie ryzyka (zagrożeń) oraz jego ocenę. W połączeniu ze zdefiniowaniem wszelkich wymaganych środków ograniczających ryzyko proces ten jest często nazywany oceną ryzyka i staje się najlepszą praktyką w zakresie wydajnych programów bezpieczeństwa maszyn

3.1 Normy regulujące bezpieczeństwo pracy stanowisk zrobotyzowanych

Kwestie bezpieczeństwa pracy na zrobotyzowanych stanowiskach są uregulowane na całym świecie. Wszędzie obowiązują standardy bezpieczeństwa związane z robotami przemysłowymi. W USA jest to ANSI/RIA R15.06-2012 („American National Standard for Industrial Robots and Robot Systems-Safety Requirements”, American National Standards Institute, New York, 2012), który odpowiada międzynarodowemu standardowi ISO 10218-1 („Robots and robotic devices – Safety requirements for industrial robots” – Part 1: Robots) oraz ISO 10218-2 („Robots and robotic devices – Safety requirements for industrial robots” – Part 2: „Robot systems and integration”). Standard ANSI jest podstawą standardu kanadyjskiego CAN/CAS-Z434-03 (R2013) („Industrial

Robots and Robot Systems – General Safety Requirements", Canadian Standards Association, Toronto, Canada, 2013). Z kolei w Japonii obowiązują standardy JIS („An Interpretation of the Technical Guidance on Safety Standards in the Use, etc., of Industrial Robots", Japanese Industrial Safety and Health Association, Tokio, 1985).

Wszystkie kraje członkowskie Unii Europejskiej są zobowiązane do wprowadzenia przepisów prawnych określających zasadnicze wymagania bezpieczeństwa dotyczące maszyn oraz ich użytkowania. Normy zharmonizowane zawierające wymagania dotyczące bezpieczeństwa maszyn są związane z dyrektywą 2006/42/WE – Bezpieczeństwo maszyn. Poprzednia wersja - 98/37/EC - została uzupełniona pod koniec 2009 r. Wprowadzono w niej pewne zmiany, uwzględniające rozwój technologii i metodologii. Od producenta lub autoryzowanego przedstawiciela wymaga się:

- zapewnienia zastosowania odpowiednich środków BHP wyszczególnionych w załączniku I dyrektywy,
- przygotowania dokumentacji technicznej,
- przeprowadzenia odpowiedniej oceny zgodności,
- udostępnienia „Deklaracji zgodności EC",
- dołączenia oznaczenia CE tam, gdzie jest ono wymagane,
- zapewnienia instrukcji umożliwiających bezpieczne użytkowanie.

W dyrektywie narzędziowej 2009/104/WE (dawniej 89/656/EWG) określono obowiązki pracodawcy. Obejmuje ona kwestie użytkowania maszyn oraz urządzeń w miejscu pracy. Nakłada na pracodawcę obowiązek stosowania się do przepisów związanych z użyciem sprzętu roboczego. Wprowadza się w niej również szeroko rozumiane podwyższenie poziomu bezpieczeństwa oraz ochrony zdrowia. Ponadto, każdy kraj przyjmujący tę dyrektywę ma możliwość dodania własnych wymagań.

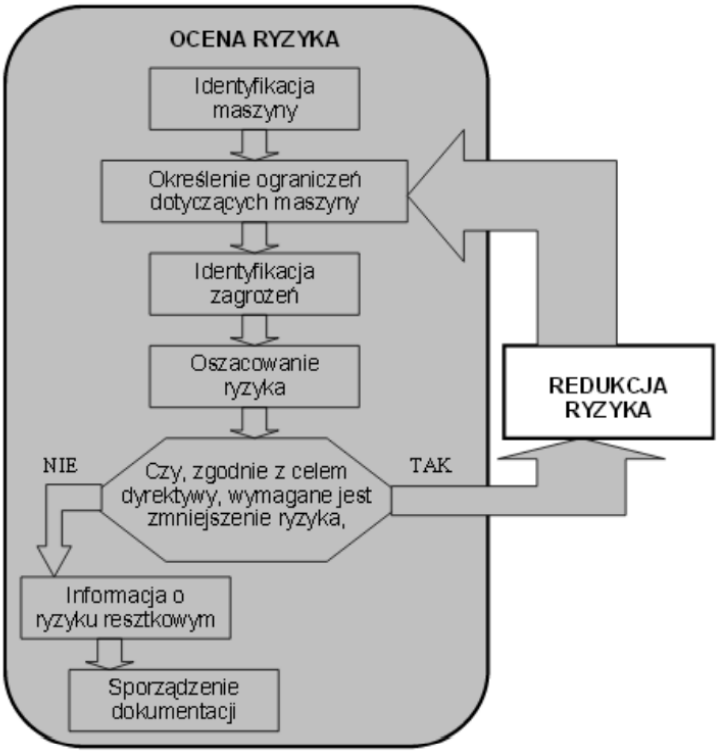
W tabeli 1 przedstawiono wybrane normy dotyczące bezpieczeństwa maszyn. W normach tych podane są m.in. ogólne zasady projektowania maszyn i związane z tym wymagania oraz zasady doboru środków ochronnych.

Tabela 1. Wybrane normy dotyczące maszyn i urządzeń

Nr normy	Tytuł normy
PN-EN ISO 12100:2012	Bezpieczeństwo maszyn – Ogólne zasady projektowania – Ocena ryzyka i zmniejszanie ryzyka

PN-EN 349+A1:2010	Bezpieczeństwo maszyn – Minimalne odstępy zapobiegające zgnieceniu części ciała człowiek
PN-EN 547-1+A1:2010	Bezpieczeństwo maszyn – Wymiary ciała ludzkiego – Część 1: Zasady określania wymiarów otworów umożliwiających dostęp całym ciałem do maszyny
PN-EN ISO 13857:2010	Bezpieczeństwo maszyn – Odległości bezpieczeństwa uniemożliwiające sięganie kończynami górnymi i dolnymi do stref niebezpiecznych
PN-EN 981+A1:2010	Bezpieczeństwo maszyn – System dźwiękowych i wizualnych sygnałów niebezpieczeństwa oraz sygnałów informacyjnych
PN-EN ISO 7731:200	Ergonomia – Sygnały bezpieczeństwa dla obszarów publicznych i obszarów pracy – Dźwiękowe sygnały bezpieczeństwa
PN-EN 842+A1:2010	Bezpieczeństwo maszyn – Wizualne sygnały niebezpieczeństwa – Ogólne wymagania, projektowanie i badanie
PN-EN ISO 14120:2016-03	Bezpieczeństwo maszyn – Osłony – Ogólne wymagania dotyczące projektowania i budowy osłon stałych i ruchomych
PN-EN ISO 13849-1:2016-02	Bezpieczeństwo maszyn – Elementy systemów sterowania związane z bezpieczeństwem – Część 1: Ogólne zasady projektowania
PN-EN 1037+A1:2010	Bezpieczeństwo maszyn – Zapobieganie niespodziewanemu uruchomieniu
PN-EN ISO 12100:2012	Bezpieczeństwo maszyn – Ogólne zasady projektowania – Ocena ryzyka i zmniejszanie ryzyka
PN-EN ISO 14119:2014-0	Bezpieczeństwo maszyn – Urządzenia blokujące sprzężone z osłonami – Zasady projektowania i doboru
PN-EN 894-1+A1:2010	Bezpieczeństwo maszyn – Wymagania ergonomiczne dotyczące projektowania wskaźników i elementów sterowniczych – Część 1: Ogólne zasady interakcji między człowiekiem a wskaźnikami i elementami sterowniczymi
PN-EN 894-2+A1:2010	Bezpieczeństwo maszyn – Wymagania ergonomiczne dotyczące projektowania wskaźników i elementów sterowniczych – Część 2: Wskaźniki
PN-EN 1037+A1:2010	Bezpieczeństwo maszyn – Zapobieganie niespodziewanemu uruchomieniu
PN-EN ISO 10218-1:2011	Roboty i urządzenia dla robotyki – Wymagania bezpieczeństwa dla robotów przemysłowych – Część 1: Roboty
PN-EN ISO 10218-2:2011	Roboty i urządzenia dla robotyki – Wymagania bezpieczeństwa dla robotów przemysłowych – Część 2: System robotowy i integracja

Normy PN-EN ISO 10218-1:2011 „Roboty i urządzenia dla robotyki. Wymagania bezpieczeństwa dla robotów przemysłowych" (część 1: „Roboty" oraz część 2: „System robotowy i integracja") są normami szczegółowymi skierowanymi przede wszystkim do producentów robotów. W części pierwszej zawarto zalecenia dotyczące bezpiecznego projektowania, prawidłowego wyboru środków ochronnych, a także informacje dla użytkowników na temat wykorzystywania robotów przemysłowych. W części tej wymieniono zagrożenia związane z pracą z robotami oraz zalecenia mające na celu ich eliminację lub zmniejszenie. Ponadto zamieszczono w niej informacje na temat poprawnej weryfikacji i walidacji systemów bezpieczeństwa.



Rys. 1. Ogólna metodyka oceny ryzyka wg PN-EN ISO 10218-1:2011
(Krupa P., Gawłowicz P. Komputerowe wspomaganie projektowania bezpiecznego stanowiska pracy robota DOI: 10.21008/j.0239-9415.2017.072.09)

Część druga normy ISO 10218 jest rozszerzeniem części pierwszej, dotyczącej pojedynczych robotów, i dotyczy całego ich systemu. Skierowano ją do instalatorów systemów zrobotyzowanych i użytkowników.

Według dyrektywy (2006/42/WE) istnieje obecnie wyraźny wymóg oceny ryzyka w celu ustalenia, które wymagania BHP są odpowiednie. Zostały również wprowadzone zmiany w procedurach oceny zgodności sprzętu z załącznika IV. Dokładna analiza ryzyka to kluczowy etap początkowy w zarządzaniu ryzykiem. Obejmuje pomiar dwóch wartości ryzyka: wartości potencjalnej szkody oraz prawdopodobieństwa, że taka szkoda może wystąpić. Na rysunku 1 przedstawiono ogólną metodykę oceny ryzyka.

Sprawny proces identyfikacji ryzyka polega na badaniu działań pracownika oraz ryzyka, które jest związane z jego pracą lub na jakie może on narazić obiekt z powodu niewystarczającego przeszkolenia lub niewielkiego doświadczenia. Analiza ryzyka powinna również obejmować identyfikację ryzyka pracowników i urządzeń w zakładzie oraz dla środowiska wynikającego z możliwości narażenia środowiskowego lub z ograniczonych zabezpieczeń w przypadku nieprawidłowej instalacji lub awarii urządzeń. Wielkość potencjalnej szkody jest sprawdzana w zakresie od najgorszego scenariusza, prowadzącego do zgonu człowieka i przestoju maszyny, do scenariusza najbardziej pozytywnego, obejmującego zachowania niebezpieczne oraz obniżenie poziomów produkcji. Na tej podstawie obliczane jest prawdopodobieństwo powstania danej szkody.

4 Podstawowe informacje o programie RobotStudio

Program RobotStudio służy do programowania robotów ABB w trybie off-line (bez fizycznego połączenia z robotem). W programie można tworzyć całe cele zawierające robota przemysłowego, urządzenia transportowe i pomocnicze, oraz materiały (np. elementy obrabiane, pakowane, montowane lub poddawane innym procesom). Praca robota przemysłowego oraz wszystkich urządzeń w zaprojektowanej celi może być symulowana w środowisku wirtualnym. Dzięki tym funkcjom, RobotStudio może służyć jako narzędzie do przygotowania projektów stanowisk zrobotyzowanych dla wielu zastosowań.

4.1 Instalacja i licencjonowanie

Program RobotStudio można pobrać bezpośrednio ze strony ABB Robotics:

https://new.abb.com/products/robotics/robotstudio

Instalacja programu jest łatwa. Pobrany plik należy rozpakować a następnie uruchomić program "setup.exe" znajdujący się w podkatalogu "RobotStudio". Później postępować według poleceń na ekranie.

Przez pierwsze 30 dni po instalacji program działa z licencją próbną, później licencję można przedłużyć kontaktując się z opiekunem Laboratorium Automatyki Przemysłowej (R210).

Procedury połączenia z serwerem i przedłużenia okresu ważności licencji opisane są w rozdziale ***Rozwiązywanie problemów z RobotStudio***.

4.2 Dodatki do programu RobotStudio

4.2.1 Sterowniki wirtualne RobotWare

Moduły RobotWare zawierają kompletne oprogramowanie sterowników robotów ABB oraz narzędzia pozwalające na uruchomienie sterowników wirtualnych w pamięci komputera.
Po zainstalowaniu modułów RobotWare możliwe będzie utworzenie cyfrowego (wirtualnego) obrazu celi z robotem, przeprowadzenie symulacji pracy robota oraz sprawdzenie poprawności programu na robota (programu, który później może być wysłany do rzeczywistego sterownika robota).

Moduły RobotWare zawierają wszystkie funkcje, w które wyposażona jest grupa robotów obsługiwanych przez wybrany sterownik.

Najnowsza wersja RobotWare 7 zawiera oprogramowanie sterowników OnmiCore, które sterują robotami:

a) IRB 1100 - mały robot szeregowy (6 osi, udźwig 4kg)
b) IRB 1300 - kompaktowy robot uniwersalny (szeregowy; 6 osi, udźwig do 11 kg)
c) IRB 360 - FlexPicker; robot typu delta (udźwig do 8kg w zależności od wersji)
d) IRB 910INV - robot typu SCARA (udźwig do 3 kg)
e) CRB 1100 - szeregowy robot kolaboracyjny (udźwig 4kg)
f) CRB 15000 - szeregowy robot kolaboracyjny (udźwig 5kg)
g) IRB 14050 - Yumi (jedno ramię) - szeregowy robot kolaboracyjny (udźwig 0.5kg)

Wersje RobotWare 6 i 5 zawierają sterowniki instalowane we wcześniej produkowanych robotach. Należy je zainstalować aby w RobotStudio projektować stanowiska z robotami, które nie znajdują się na liście powyżej.
RobotWare można pobrać ze strony ABB lub z biblioteki dodatków dostępnej w programie RobotStudio.

4.2.2 Rozszerzenia PowerPacs

Do programu RobotStudio można dodać rozszerzenia PowerPacs, które zwiększają jego funkcjonalność w zakresie projektowania wybranych procesów. Na stronie ABB dostępne są następujące dodatki PowerPacs:

a) ArcWelding i ArcWelding 2 - do projektowania procesów spawania, w których narzędzie spawalnicze prowadzone jest przez robota,
b) Cutting - do projektowania procesów cięcia (2D i 3D),
c) Machining - do projektowania procesów obróbki skrawaniem (np. frezowanie, gratowanie, szlifowanie, polerowanie),
d) Machine Tending - do projektowania zrobotyzowanej obsługi maszyn cnc w celi produkcyjnej
e) Painting - do projektowania procesów malowania z użyciem robotów,
f) Palletizing - do projektowania procesów pakowania i paletyzacji,
g) Picking - do projektowania procesów śledzenia, grupowania i wybierania elementów (np. podnoszenie wyrobów z transportera i wkładanie ich do opakowań detalicznych),
h) Dispensing - do projektowania procesów nakładania (np. kleju, masy uszczelniającej) na przygotowane powierzchnie elementów,
i) 3D Printing - do projektowania procesów druku 3D (robot prowadzi głowicę drukującą) .

Dodatki PowerPacs zawierają dodatkowe zestawy instrukcji oraz procedur przydatnych w projektowaniu i programowaniu stanowisk robotów przeznaczonych do realizowania wymienionych wyżej procesów. Należy jednak zaznaczyć, że instalowanie dodatków rekomendowane jest tylko wtedy, gdy robot będzie wykonywał zadania typowe dla procesów do których przeznaczony jest PowerPac. Efektem ubocznym użycia funkcji zawartych w dodatkach może być ograniczenie lub zablokowanie niektórych funkcji robota w projektowanej celi.

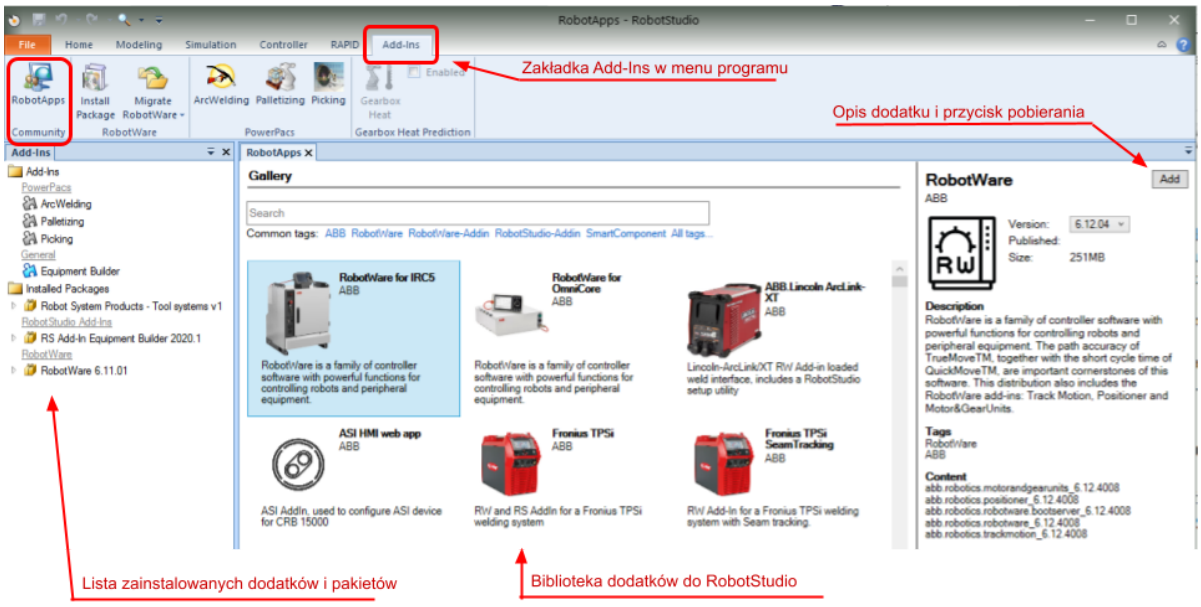
4.2.3 Biblioteka modułów RobotApps

Z aplikacją RobotStudio powiązana jest biblioteka RobotApps, dostępna w internecie pod adresem:

https://robotapps.robotstudio.com

oraz bezpośrednio w programie (zakładka Add-Ins, Rys. 2).

Z biblioteki można pobrać sterowniki RobotWare, moduły urządzeń (na rys. 1 widoczne są moduły sterowników spawalniczych), modele geometryczne urządzeń (ogrodzenia, podesty pod roboty, itp.), mechanizmy (narzędzia malarskie i spawalnicze, chwytaki mechaniczne, chwytaki podciśnieniowe, itp.), dodatki do konfiguracji dodatkowych osi i innych urządzeń, modele stanowisk zrobotyzowanych, modele maszyn i inne. Biblioteka ma charakter otwarty, poza elementami opublikowanymi przez ABB, znajdują się w niej elementy umieszczone przez użytkowników programu RobotStudio.



Rys. 2. Biblioteka RobotApps w programie RobotStudio. W oknie po lewej stronie znajduje się lista zainstalowanych elementów

4.3 Pliki projektu w RobotStudio

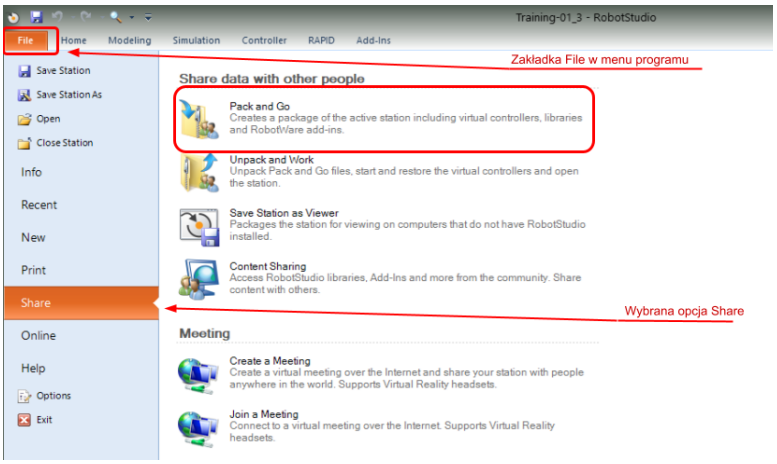
4.3.1 Struktura plików w projekcie

RobotStudio zapisuje projekty w podkatalogu ***Solutions*** wewnątrz katalogu ***RobotStudio*** tworzonego domyślnie w Dokumentach w katalogu użytkownika Windows.
Nazwa katalogu z projektem jest ustalana w trakcie tworzenia nowego projektu (domyślnie jest to ***SolutionNN*** - gdzie NN jest kolejnym numerem projektu np. Solution23). Folder projektu zawiera

szereg podkatalogów tworzonych automatycznie w czasie pracy z projektem (zwykle tworzone są katalogi:

a) Backups - zawiera kopie zapasowe sterowników robotów użytych w projekcie (kopie są domyślnie tworzone automatycznie),
b) Libraries - pliki bibliotek związane z projektem,
c) RAPID Programs - programy w języku RAPID (język programowania robotów ABB),
d) SignalAnalyzer - zarejestrowane przebiegi sygnałów (narzędzie do rejestracji pracy robota i urządzeń peryferyjnych w czasie symulacji),
e) Stations - pliki zawierające zapis struktury stacji (stanowiska) oraz ich kopie zapasowe,
f) Virtual Controllers - pliki wirtualnych sterowników użytych w projekcie.

Do katalogu Stations domyślnie zapisywane są również pliki .rspag (do archiwizacji lub przenoszenia projektu - opis w kolejnym rozdziale).



Rys. 3. Menu archiwizacji i prezentacji projektu w RobotStudio

4.3.2 Prezentacja projektu, archiwizacja i przenoszenie projektów

Projekt wykonany w RobotStudio można zarchiwizować lub przenieść na inny komputer korzystając z funkcji **Pack and Go** (Rys. 3). Funkcja zapisuje cały projekt do jednego, skompresowanego pliku z rozszerzeniem ***.rspag***, który jest domyślnie zapisywany w podkatalogu ***Stations*** projektu.
Menu Share (rys. 3) zawiera jeszcze inne funkcje:

a) Unpack and Work - służy do odczytu (rozpakowania) archiwów ***.rspag*** utworzonych przez funkcję Pack and Go - funkcja odtwarza strukturę katalogów projektu (domyślnie w katalogu ***RobotStudio\Stations*** w Dokumentach użytkownika Windows).
UWAGA: Funkcja nie nadpisuje istniejącego katalogu - jeżeli docelowy katalog będzie już istniał na dysku proces rozpakowywania zostanie zatrzymany - należy wtedy zmienić nazwę katalogu docelowego.
b) Save Station as Viewer - zapisuje model 3D projektu oraz program symulacji (o ile istnieje w projekcie) do pliku wykonawczego (***.exe***). Plik może być uruchomiony na komputerze, na którym nie ma zainstalowanego RobotStudio. Ta forma prezentacji projektu może być wykorzystana do przedstawienia klientowi, na konferencji lub seminarium, lub do udokumentowania postępów w projekcie w trakcie zajęć na uczelni. Model zapisany w tym formacie działa również z goglami VR.
c) Content Sharing - pozwala opublikować projekt lub jego część w bibliotece RobotApps.
d) Create a Meeting - pozwala zorganizować spotkanie on-line, na którym wszyscy uczestnicy będą mieli dostęp do projektu otwartego w RobotStudio. Uczestnicy spotkania mogą oglądać projekt również poprzez gogle VR.
e) Join a Meeting - pozwala dołączyć do spotkania on-line i obejrzeć projekt otwarty na komputerze gospodarza (organizatora) spotkania.

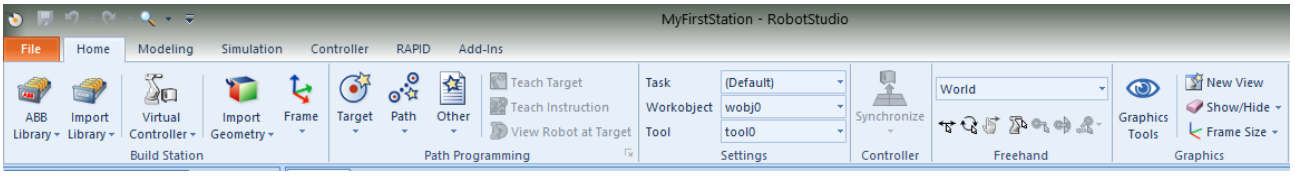
4.3.3 Import z programów CAD

Do projektu wykonanego w RobotStudio można importować modele 3D z programów CAD. Modele 3D powinny być zapisane w formacie:

- **ACIS** (pliki z rozszerzeniami **.sat**, .sab, .asat, .asab) - format wewnętrzny programu,
- innym (CATIA, DXF/DWG, IGES, Inventor, JT, LDraw, NX, Parasolid, Solid Edge, Solid Works, STEP, VRML, VRML2) - wymagana jest odpowiednia licencja.

4.3.4 Ogólne zasady pracy z programem RobotStudio

Przyciski na paskach menu w programie RobotStudio ułożone są w taki sposób, że większość procedur wykonuje się używając przycisków kolejno - zaczynając od lewej strony i przesuwając się w prawo.



Rys. 4. Pasek menu w zakładce Home

Tworząc projekt w kolejnych krokach będziemy używać przycisków: ABB Library, Import Library, Virtual Controller, Import Geometry, Frame, Target, Path , Other (Rys. 4) w takiej właśnie kolejności.
Pierwszym krokiem będzie utworzenie nowego projektu przy pomocy funkcji z menu File.

5 Tworzenie projektu stanowiska robota

W tym przewodniku utworzony będzie proste stanowisko robota z podstawowym narzędziem. Zadaniem robota będzie przeprowadzić narzędzie po ścieżce wyznaczonej przez krawędzie części.
Przed przystąpieniem do pracy należy:

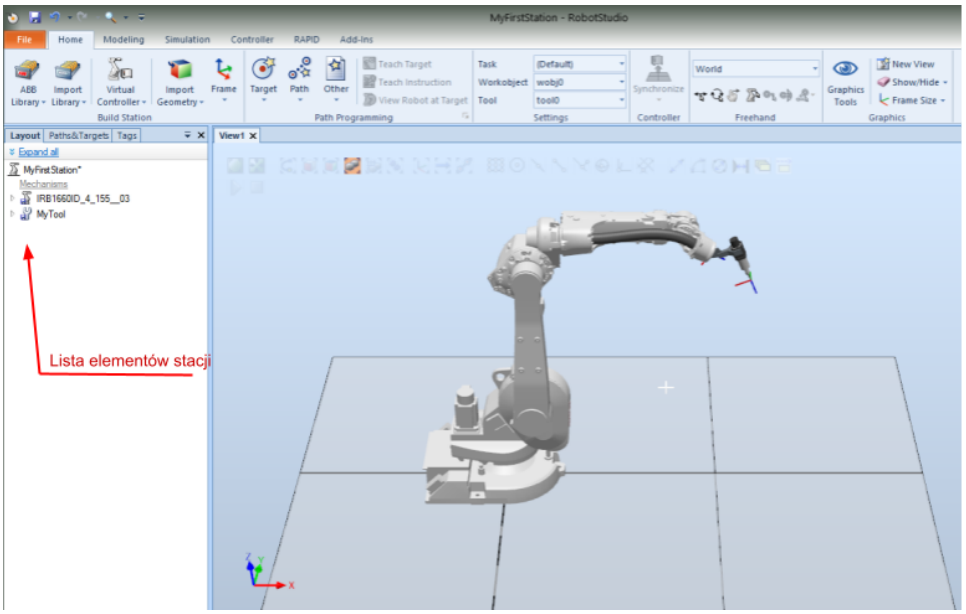
a) zainstalować RobotStudio i pobrać licencję,
b) zainstalować RobotWare 6,
c) połączyć komputer z internetem (będziemy korzystać z biblioteki RobotApps).

5.1 Tworzenie modelu celi robota

Aby utworzyć pustą przestrzeń, która będzie wypełniana elementami stanowiska z robotem wykonamy kolejno kroki:

1) z menu **File** wybieramy opcję **New** i **Solution with Empty Station**,
2) po wpisaniu nazwy projektu (np. ***MyFirstStation***) wciskamy przycisk **Create**, pojawi się ekran projektu z zaznaczonym układem współrzędnych i pustą podłogą,
3) po utworzeniu projektu program automatycznie uaktywni pasek menu w zakładce **Home** (rys. 3).
4) używamy przycisku **ABB Library** aby otworzyć bibliotekę robotów ABB, wybieramy robota IRD1660ID - cyfrowy model robota pojawi się na podłodze,

5) pod przyciskiem **Import Library** znajduje się biblioteka narzędzi i akcesoriów - z grupy **Equipment** wybieramy **Training Objects** (trzeba przewinąć listę do góry - sekcja znajduje się na samym dole listy) i klikamy w narzędzie **My Tool**, narzędzie pojawi się na ekranie ale będzie widoczne tylko w zarysach ponieważ zostanie umieszczone dokładnie w tym samym miejscu co podstawa robota - w punkcie o współrzędnych [X=0, Y=0, Z=0] głównego układu współrzędnych związanego z projektem,
6) aby umieścić narzędzie na flanszy robota należy w lewym oknie programu zawierającym listę elementów celi (zakładka ***Layout***) przeciągnąć narzędzie MyTool i upuścić go na robota (IRB1660ID_4_155__03), należy odpowiedzieć **Yes** na pojawiające się zapytanie "Czy chcesz zaktualizować pozycję MyTool?". Po tej operacji cela powinna wyglądać jak na rys. 5.



Rys. 5. Cela po dodaniu robota i narzędzia

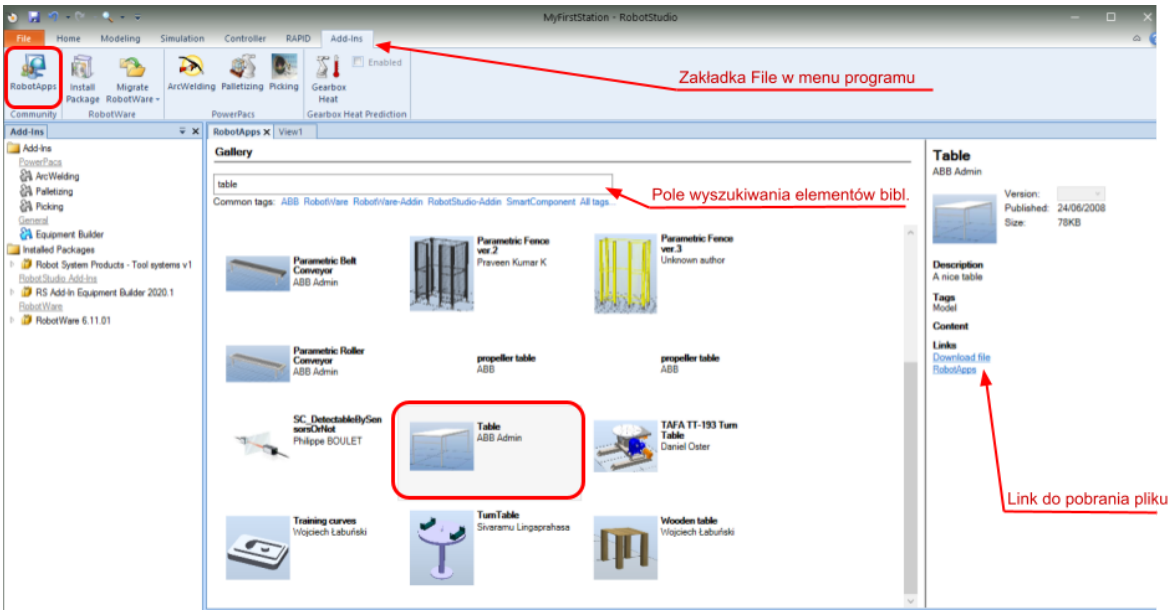
Zadanie 1: Proszę odłączyć narzędzie od robota, wybrać inne z biblioteki programu i umieścić je na robocie. Następnie na powrót umieścić narzędzie ***MyTool*** na robocie.

Wskazówka: Aby odłączyć narzędzie od robota należy skorzystać z funkcji ***Detach*** dostępnej w menu pojawiającym się po kliknięciu prawym klawiszem na narzędzie w liście elementów stacji.

5.2 Import elementów wyposażenia z biblioteki RobotApps

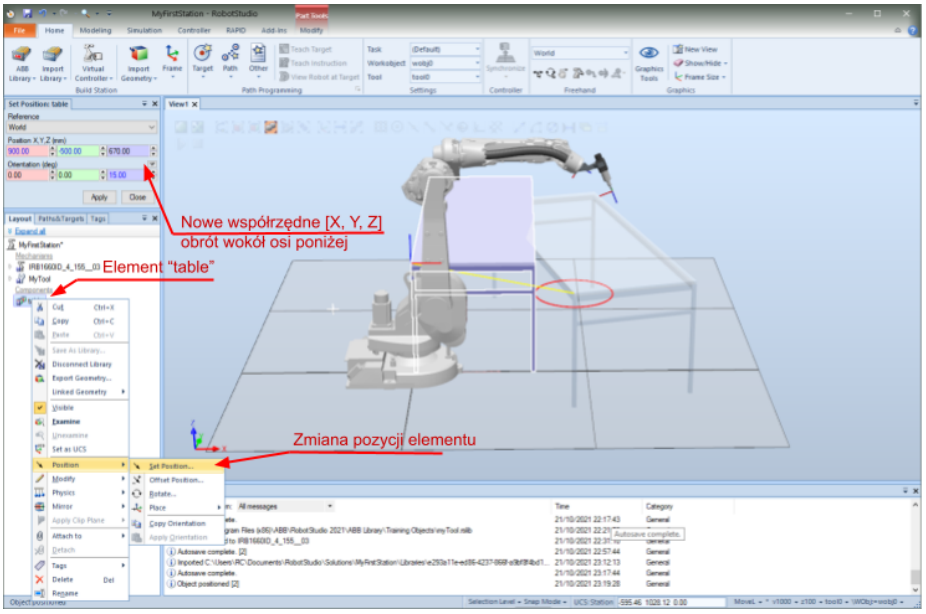
W kolejnych krokach do tworzonej celi wstawimy modele urządzeń z biblioteki RobotApps. Przedmiot "obrabiany przez robota będzie leżał na stole, robot zostanie postawiony na podeście. Należy wykonać następujące czynności:

7) wybrać zakładkę **Add-Ins** i wybrać **RobotApps** (pierwsza pozycja z lewej) na pasku menu - pojawi się okno z zawartością biblioteki,
8) w pole wyszukiwania (w górnej części okna biblioteki) wpisujemy ***table*** - po wciśnięciu Enter w katalogu zostaną tylko elementy zawierające słowo "table" w nazwie lub w opisie,
9) wybieramy stół "Table" opublikowany przez "ABB Admin" i używamy odnośnika (linku) "Download file" w prawym oknie (okno z opisem elementu) do pobrania pliku z modelem stołu (rys. 6). Pobrany plik (z rozszerzeniem ***.rslib***) należy umieścić w katalogu **Libraries** projektu.



Rys. 6. Pobieranie modelu stołu z biblioteki RobotApps

10) Po zapisaniu pliku w podkatalogu **Libraries** naszego projektu (np. ***RobotStudio\MyFirstStation\Libraries***) model stołu pokaże się w menu **Home → Import Library → Solution Library**,
11) model stołu wstawiamy do projektu wybierając go z menu Solution Library,
12) stół pojawi się w projekcie ustawiony jedną nogą w punkcie [0, 0, 0], aby go ustawić w innym miejscu należy kliknąć prawym klawiszem myszy w element "**table**" w lewym oknie programu i z rozwiniętego menu wybrać **Position → Set Position** (rys. 7),
13) w oknie, które się pojawi, wpisujemy nowe współrzędne stołu [X=900, Y=-500, Z=670] oraz dodatkowo obrócimy stół o 15 stopni wokół osi Z [Rx=0, Ry=0, Rz=15], po wciśnięciu **Apply** stół ustawi się w nowej pozycji (rys. 7).



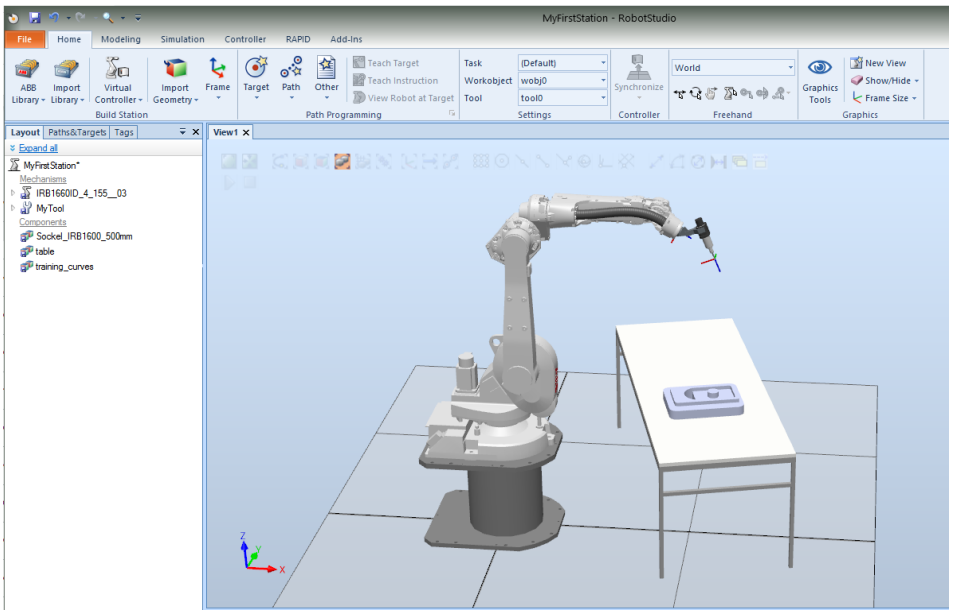
Rys. 7. Zmiana pozycji stołu w projektowanej celi

W podobny sposób z biblioteki RobotApps pobierzemy model części **Training Curves** oraz model podstawy pod robota.

14) część **Training Curves** opublikowana przez Wojciecha Łabuńskiego jest już widoczna w bibliotece (rys. 6) - należy pobrać plik i umieścić go w katalogu Libraries projektu (tak samo jak model stołu),

15) pozycję części **Training Curves** należy ustawić na [X=900, Y=0, Z=670] oraz [Rx=90, Ry=0, Rz=15],
16) następnie należy pobrać z biblioteki podstawę robota **IRB1600 Base 500 mm** i również umieścić ją w katalogu Libraries projektu a następnie wstawić do projektu.

Zadanie 2: Umieścić robota na podstawie zaimportowanej do projektu.
Wskazówka: Ponieważ nasz robot stoi w punkcie [0, 0, 0] umieszczenie go na podstawie będzie łatwe - wystarczy jedynie zmienić współrzędną Z robota z Z=0 na Z=500 w podobny sposób jak w przypadku stołu i części na stole (rys. 8).



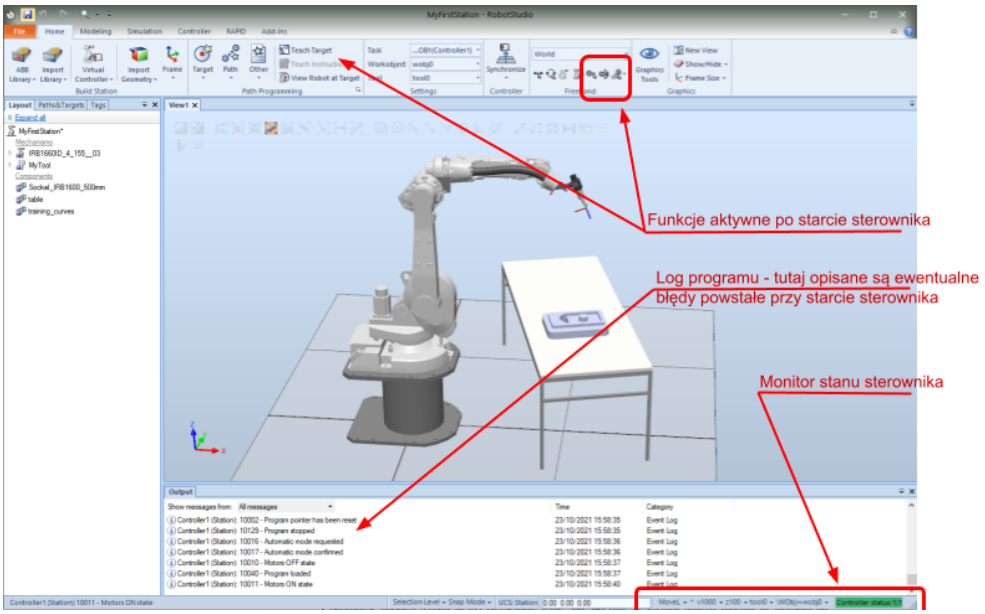
Rys. 8. Wygląd celi po wstawieniu elementów z biblioteki RobotApps

5.2.1 Dodawanie sterownika robota

Kolejnym krokiem w projekcie jest dołączenie i uruchomienie sterownika robota. Dzięki tej operacji zyskamy możliwość sterowania robotem nie tylko w układzie współrzędnych związanym z maszyną (zmiana pozycji przegubów) ale również w układzie kartezjańskim (ustawienie narzędzia w punkcie o współrzędnych [X,Y,Z] i w określonej orientacji [Rx,Ry,Rz]). Po zainstalowaniu sterownika będzie również możliwe programowanie robotem.

Najprostszą metodą dołączenia sterownika jest skorzystanie z funkcji **From Layout…** znajdującej się pod przyciskiem **Virtual Controller**. Funkcja automatycznie dobierze sterownik w konfiguracji dostosowanej do urządzeń, które znajdują się w naszym projekcie.

17) Po wybraniu funkcji (***Virtual Controller → From Layout ...***) pojawi się okno dialogowe z nazwą sterownika (zwykle "*Controller 1*") i wybraną wersją oprogramowania RobotWare (dla robota IRB1660 będzie to RobotWare 6). Należy wcisnąć **Finish** i poczekać aż sterownik zostanie skonfigurowany i uruchomiony.



Rys. 9. Okno programu po prawidłowym zainstalowaniu i uruchomieniu sterownika

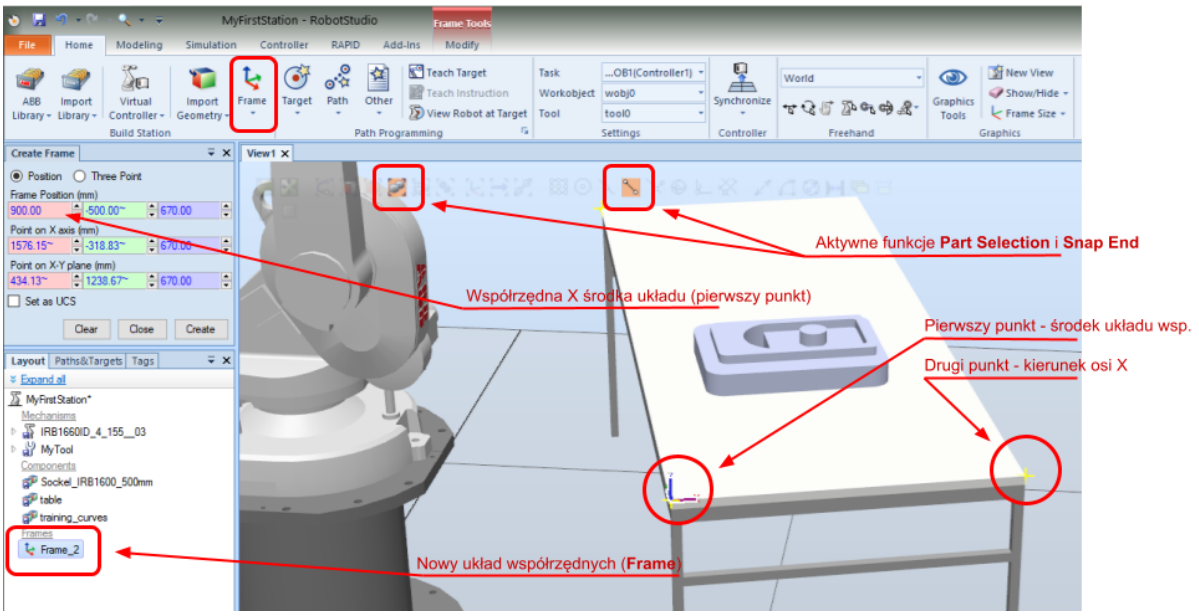
Po prawidłowym zainstalowaniu i uruchomieniu sterownika dostępna będzie funkcja sterowania położeniem narzędzia w układzie kartezjańskim (**Jog**) a w dolnej części okna programu pojawi się wskaźnik stanu sterownika (rys. 9).

Uwaga: *W razie problemów z dołączeniem sterownika należy sprawdzić czy w RobotStudio zainstalowana jest odpowiednia wersja* ***RobotWare****.*

Uwaga: *Funkcja automatycznego doboru i dołączania sterownika (**From Layout ...**) dostępna jest tylko z aktywną licencją Premium - jeżeli ta funkcja nie jest dostępna w menu, należy połączyć się z serwerem licencji lub wypożyczyć licencję czasową (Commuter License).*

5.3 Definiowanie obszaru roboczego

Zanim zaczniemy definiować punkty robocze (targets) i projektować ścieżki narzędzia konieczne jest zdefiniowanie przynajmniej jednego układu odniesienia związanego z obszarem roboczym. W RobotStudio obszar roboczy nazywa się **workobject**. Obszar może być związany bezpośrednio z przedmiotem manipulowanym (lub obrabianym) przez robota, z jednym z urządzeń w celi robota (np. z transporterem) lub z innym elementem wyposażenia celi. Istotne jest aby zdefiniowany układ odniesienia był łatwy do znalezienia (wyznaczenia) w rzeczywistej celi robota.



Rys. 10. Definiowanie nowego układu współrzędnych przy pomocy trzech punktów na krawędzi stołu

W jednej celi robota może być zdefiniowanych kilka obszarów roboczych. Na przykład w celi paletyzacji można zdefiniować obszar roboczy związany z transportem z którego odbierane są przedmioty do pakowania i drugi, związany z paletą na którą przedmioty mają być odkładane. W ten sposób po zbudowaniu rzeczywistej celi wystarczy wpisać do sterownika robota dwa punkty odniesienia (definiujące układy współrzędnych związane z obszarami roboczymi) i robot będzie mógł zacząć pracę. Rozdzielenie obszarów roboczych (pobierania i odkładania) ułatwi konfigurację całej celi ponieważ położenie palety nie będzie związane z położeniem transportera odbiorczego -

ewentualne różnice odległości (w stosunku do projektu w RobotStudio) zostaną automatycznie uwzględnione przez sterownik robota.

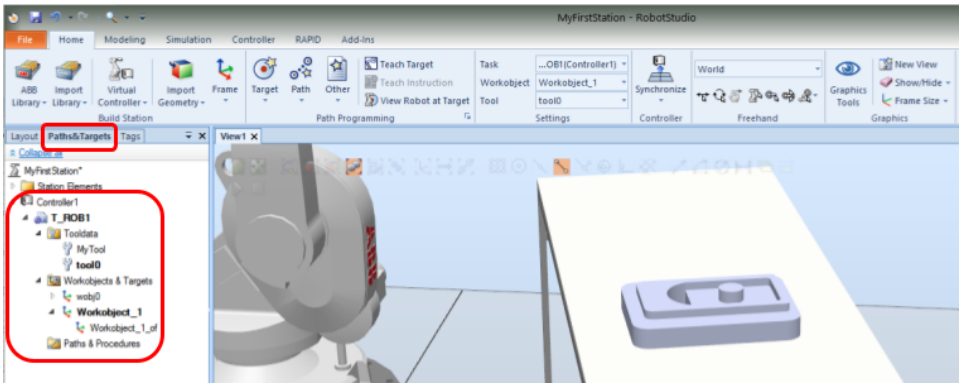
W projekcie utworzymy obszar roboczy związany ze stołem. Początek lokalnego układu współrzędnych związanego z nowym obszarem roboczym będzie znajdował się na rogu stołu. Należy wykonać następujące kroki:

18) w oknie widoku projektu wybrać ***Part Selection*** i ***Snap End*** (por. rys. 10),
19) z menu **Frame** wybrać **Frame from Three Points** - z lewej strony ekranu pojawi się okno do wpisania współrzędnych - należy uaktywnić pierwsze pole ***Frame Position*** (do wpisania współrzędnej X, czerwone) klikając w nie wskaźnikiem myszy,
20) najechać wskaźnikiem myszy w okolice najbliższego rogu stołu - na końcu krawędzi stołu (w rogu stołu) pojawi się biała kulka oznaczająca wybrany punkt - po kliknięciu lewym przyciskiem myszy współrzędne punktu zostaną wpisane do *Frame Position*,
21) uaktywnić pole do wpisania współrzędnej X drugiego punktu *Point on X Axis* - postępując podobnie jak poprzednio wskazać myszą punkt na krótszej krawędzi stołu - będzie to kierunek osi X definiowanego układu współrzędnych,
22) postępując podobnie jak wyżej, zaznaczyć pole X trzeciego punktu (*Point on X-Y plane*) i wskazać punkt na dłuższej krawędzi stołu,
23) wcisnąć **Create** - w liście ***Layout*** pojawi się symbol nowo utworzonego układu współrzędnych (np. ***Frame_1***).

Ostatnim krokiem tej procedury jest utworzenie obszaru roboczego (workobject) w którym układem odniesienia będzie utworzony powyżej ***Frame_1***. Należy:

24) kliknąć prawym klawiszem myszy na obiekt ***Frame_1*** i listy, która się pojawi wybrać **Convert frame to workobject**.

Po wykonaniu tej operacji obiekt ***Frame_1*** zostanie zabrany z listy ***Layout*** i przeniesiony do listy ***Paths&Targets*** jako **Workobject_1**. Równocześnie Workobject_1 zostanie ustawiony jako aktywny, co jest w RobotStudio sygnalizowane pogrubioną czcionką (rys. 11).



Rys. 11. Utworzony obszar roboczy workobject_1 jest zdefiniowany w strukturze sterownika robota (Controller1) w zakładce Paths&Targets

W strukturze sterownika na liście ***Paths&Targets*** znajduje się obszar pracy **wobj0**. Obszar **wobj0** jest dodawany do listy w czasie inicjalizacji sterownika - układ współrzędnych tego obszaru jest zaczepiony w tym samym miejscu gdzie znajduje się punkt zerowy podstawy robota. W obszarze **wobj0** nie należy tworzyć punktów trajektorii i ścieżek narzędzia gdyż w praktyce (w rzeczywistej celi) oznaczałoby to konieczność ustawienia wszystkich elementów celi względem punktu znajdującego się na środku podstawy robota (co może być trudne do realizacji).
W obszarze roboczym **wobj0** można zdefiniować punkty początkowe i końcowe ścieżek narzędzia - pozwala to ustawić robota w określonej pozycji (np. w pozycji ***Home*** lub w pozycji serwisowej) bez względu na położenie innych obszarów roboczych.

5.3.1 Ustawienie aktywnego narzędzia

Podstawowy wymiar geometryczny narzędzia robota zdefiniowany jest jako odległość (przesunięcie i obrót) układu współrzędnych centralnego punktu narzędzia **TCP** (w języku angielskim: Tool Centre Point) od środka układu współrzędnych związanego z flanszą robota (miejscem, do którego narzędzie jest przykręcone. Aby ta odległość została uwzględniona w przez sterownik robota w czasie pracy narzędzie musi być aktywne.
Na rys. 10 widać, że w strukturze sterownika ***Controller1*** są zapisane dwa narzędzia:

- tool0 - narzędzie domyślne o długości zero (na rys. 11 zaznaczone pogrubioną czcionką)
- MyTool - narzędzie dodane do robota w czasie tworzenia celi.

Domyślnie narzędziem aktywnym jest ***tool0*** które zawsze jest dodawane do struktury sterownika w czasie jego konfiguracji. Robot z aktywnym narzędziem ***tool0*** zachowuje się tak jakby żadnego

narzędzia nie było (narzędzie ma długość zero). Aby zmienić to zachowanie, należy uaktywnić narzędzie ***MyTool***. Aktywacja narzędzia jest prosta - wystarczy:

25) kliknąć prawym klawiszem myszy na narzędziu ***MyTool*** w strukturze sterownika robota i wybrać funkcję **Set as active**

Teraz punktem końcowym robota będzie środek układu współrzędnych (***TCP***) narzędzia ***MyTool***.

5.3.2 Tworzenie ścieżki z punktów

Punkty, które później mogą być wykorzystane do wyznaczenia ścieżki narzędzia (lub punktu końcowego robota), można zdefiniować na jeden z trzech sposobów:

- ustawiając robota w pozycji docelowej i zapamiętując jego współrzędne (***Teach Target*** - funkcja aktywna po uruchomieniu wirtualnego sterownika robota),
- wpisując współrzędne punktu bezpośrednio (***Create Target***)
- wpisując współrzędne punktu wskazanego na krzywej, krawędzi lub powierzchni (***Create Target***).

Aby dodać punkt przy pomocy funkcji **Teach Target** należy najpierw ustawić robota w pozycji docelowej, ustawić właściwy obszar roboczy (*workobject*) i użyć funkcji **Teach Target** z menu paska ***Home***. Dodamy w ten sposób punkt początkowy (i równocześnie końcowy) naszej ścieżki.
Na początku programu robot będzie ustawiał się w pozycji *Home* - w jakiej znajdował się po dodaniu go do projektu aby ta pozycja była niezależna od zdefiniowanego przez nas obszaru roboczego na stole należy:

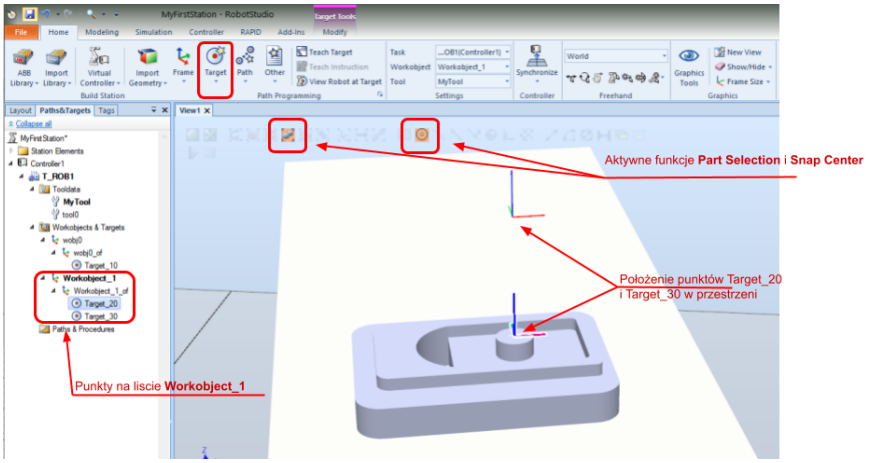
26) na liście ***Paths&Targets*** (rys. 11) ustawić jako aktywne: narzędzie **tool0** oraz obszar roboczy **wobj0** (kliknąć prawym klawiszem myszy na oba elementy i wybrać **Set as active**),
27) przejść na listę ***Layout***, kliknąć prawym klawiszem na robota (IRB1660ID) i wybrać **Move to Pose** a następnie **Home Position** - robot ustawi się w pozycji *Home*,
28) kliknąć jeden raz na **Teach Target** (w pasku menu Home, nad oknem projektu) - zostanie utworzony punkt **Target_10** i wpisany na listę **Paths&Targets** w obszarze roboczym **wobj0** (rys. 12).

Rys. 12. Target_10, został dopisany do listy w obszarze roboczym wobj0

W kolejnym kroku zdefiniujemy punkt bezpośrednio nad częścią leżącą na stole - będzie to miejsce, do którego robot będzie dojeżdżał przed rozpoczęciem pracy na stole. Punkt będzie należał do obszaru roboczego związanego ze stołem (***workobject_1***), uwzględnimy też fakt, że robot będzie miał założone narzędzie ***MyTool***. Współrzędne punktu wpiszemy ręcznie wskazując punkt na powierzchni części i modyfikując wartość współrzędnej Z. Należy:

29) na liście ***Paths&Targets*** ustawić jako aktywne: narzędzie **MyTool** oraz obszar roboczy **workobject_1** (kliknąć prawym klawiszem myszy na oba elementy i wybrać **Set as active**),
30) z paska w górnej części okna z widokiem projektu wybrać **Snap Center**,
31) z paska menu *Home* wybrać **Target** a następnie **Create Target** - po lewej stronie pojawi się okno z polami do wpisania współrzędnych,
32) w oknie **Create Target** rozwinąć listę **Reference** i wybrać **Workobject**,
33) uaktywnić pole do wpisania współrzędnej X i potem wskazać punkt na środku walca znajdującego się na części - do pól X, Y i Z wpiszą się współrzędne punktu,
34) zmienić wartość współrzędnej **Z** z 80 na 280,
35) nacisnąć **Add** a następnie **Create** i **Close**.

Po wykonaniu tych czynności na liście punktów obszaru ***workobject_1*** pojawią się punkty **Target_20 i Target_30**. W oknie projektu pojawią się dwa układy współrzędnych umieszczone w zdefiniowanych przez nas punktach (rys. 13).



Rys. 13. Punkty Target_20 i Target_30 w obszarze roboczym Workobject_1 związanym ze stołem

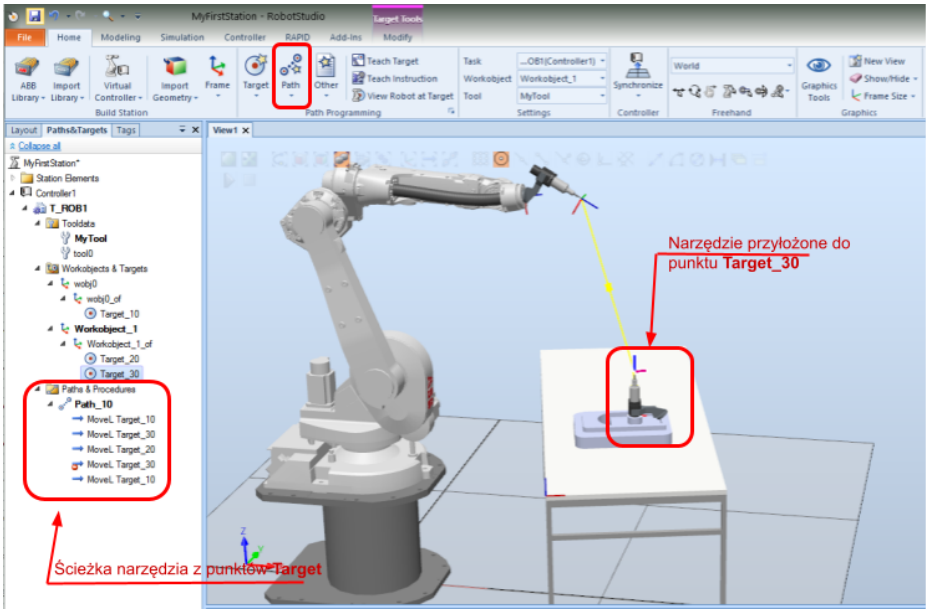
W kolejnym kroku utworzymy ścieżkę łączącą punkty **Target_10**, **Target_20** i **Target_30**. Narzędzie robota dojedzie do punktu nad częścią ruchem typu joint (każdy przegub porusza się niezależnie od innych - jest to najprostszy ruch w układzie współrzędnych robota - w języku RAPID taki ruch uruchamiany jest komendą **MoveJ**), następnie dojedzie do punktu na powierzchni części ruchem liniowym (w układzie kartezjańskim - punkt centralny narzędzia **TCP** będzie się poruszał po linii prostej do punktu docelowego), na koniec narzędzie powróci do punktu nad częścią i robot ustawi się na powrót w pozycji *Home*.
Aby utworzyć nową ścieżkę narzędzia i dodać do niej zdefiniowane punkty należy:

36) z menu **Path** (na pasku nad oknem z widokiem projektu) wybrać **Empty Path** - w strukturze sterownika zostanie utworzony obiekt ***Path_10***,
37) przeciągnąć kolejno punkty **Target_10**, **Target_30**, **Target_20** i jeszcze raz **Target_30** i **Target_10** i upuścić je na obiekt ***Path_10*** - powstanie ścieżka, którą można zobaczyć rozwijając listę ***Path_10***.

Ścieżka ***Path_10*** jest teraz zdefiniowana. Aby narzędzie robota mogło przejechać wzdłuż ścieżki należy jeszcze sprawdzić czy punkty są osiągalne (przez robota) i czy narzędzie jest ustawione w sposób właściwy.
W programie RobotStudio można sprawdzić położenie narzędzia w punkcie trajektorii (*Target)* i ustawić go tak, żeby punkt był osiągalny przez robota i narzędzie ustawione w sposób zgodny z wymaganiami technologii.

W kolejnym kroku sprawdzimy i skorygujemy położenie narzędzia w punktach ***Target_30*** i ***Target_20***. Należy:

38) kliknąć prawym klawiszem myszy na punkt **Target_30** i z rozwiniętej listy wybrać **View Tool at Target** i następnie **MyTool** - model 3D narzędzia *MyTool* zostanie przyłożony do punktu *Target_30* w sposób, jaki wynika z definicji punktu (rys. 14).



Rys. 14. Przed uruchomieniem programu należy sprawdzić osiągalność wszystkich punktów zaprojektowanej ścieżki narzędzia

W RobotStudio narzędzie przykładane jest do punktów (*Target*) w taki sposób, że układ współrzędnych przypisany do punktu *Target* i układ współrzędnych przypisany do punktu centralnego narzędzia *TCP* pokrywają się.
W naszym przypadku oznacza to, że narzędzie *MyTool* będzie przyłożone do punktu *Target_30* od dołu i zorientowane w taki sposób, że punkt będzie nieosiągalny dla robota (rys. 13). Aby skorygować położenie punktu należy:

39) kliknąć prawym klawiszem myszy na punkt **Target_30** i z rozwiniętej listy wybrać **Modify Target** a następnie **Rotate** - pojawi się okno pozwalające na wpisanie kąta obrotu,
40) w pole **Rotation** wpisać wartość 180 (stopni) i zaznaczyć oś **Y**, wcisnąć **Apply** - zmieniając orientację układu współrzędnych związanego z punktem *Target_30* zmienimy równocześnie

orientację narzędzia - teraz będzie ono przyłożone od góry i zorientowane flanszą w stronę robota,

41) w podobny sposób należy zmodyfikować punkt ***Target_20*** (uwaga - narzędzie początkowo pojawi się pod stołem),
42) wyłączyć funkcję przykładania narzędzia do punktu klikając prawym klawiszem myszy na jeden z punktów ***Target*** i wybierając **View Tool at Target → MyTool**.

Po korekcji punktów naszej ścieżki należy sprawdzić jeszcze czy są one osiągalne przez robota z narzędziem ***MyTool***, należy:

43) ustawić narzędzie **MyTool** jako aktywne (zaznaczone pogrubioną czcionką),
44) kliknąć prawym klawiszem myszy na jeden z punktów ***Target*** i z rozwiniętej listy wybrać **View Robot at Target**,
45) sprawdzić czy robot ustawia się w każdym z punktów końcowych i pośrednich ścieżki klikając na punkty ***Target*** lub na kolejne komendy definiujące ścieżkę ***Path_10*** - należy również zwrócić uwagę na ułożenie osi robota w każdej z pozycji.

Zadanie 3: Przełączyć narzędzie na **tool0** i przeprowadzić podobny test jak opisany powyżej - sformułować wnioski.

5.4 Ułożenie robota do przejazdu wzdłuż ścieżki

Pomimo iż wszystkie punkty zaprojektowanej ścieżki są osiągalne przez robota, może się okazać, że robot nie jest w stanie poprowadzić narzędzia po tej ścieżce. Przyczyną problemów może być:

- nie istnieje przekształcenie pozwalające osiągnąć punkt docelowy w bieżącym ułożeniu robota (konfiguracji ramion),
- znacząca różnica ułożenia robota (konfiguracji ramion robota) w następujących po sobie punktach trajektorii - ruch nie może być wykonany ze względu na zbyt duże prędkości lub przyspieszenia w przegubach,
- przejazd przez punkt osobliwy lub w pobliżu tego punktu - ruch nie może być wykonany ze względu na duże przyśpieszenia w przegubach.

Skuteczną metodą ograniczenia problemów związanych z ruchem osi w pobliżu położeń osobliwych robota jest ograniczenie w programie robota ruchów programowanych w przestrzeni

kartezjańskiej (np. przejazd do punktu po prostej lub po łuku). Wszystkie komendy tego typu wymagają wykonania obliczeń kinematyki odwrotnej (przeliczenia pozycji w układzie kartezjańskim na pozycję u układzie maszyny - *joint space*), których rezultatem (w okolicy punktów osobliwych) mogą być gwałtowne zmiany położenia przegubów.
Wszędzie gdzie to jest możliwe (ruch nie jest zdeterminowany przez wymagania technologii) należy planować ruchy w przestrzeni maszyny (*joint space*), gdzie ruch każdego przegubu może być wyznaczony niezależnie a synchronizacja między przegubami jest ograniczona jedynie do czasu wykonania ruchu.
W praktyce oznacza to, że komendy wymagające przekształcenia kinematyki odwrotnej takie jak

a) ***MoveL*** - ruch po prostej do punktu docelowego,
b) ***MoveC*** - ruch po łuku do punktu docelowego,

powinny być używane tylko wtedy, gdy wymaga tego technologia (np. narzędzie musi dojechać w określony sposób, musi przejechać po określonej ścieżce), Wszystkie pozostałe ruchy (np. dojazd do obszaru pracy, przejazd do innego obszaru roboczego) powinny być wykonywane z użyciem komendy:

c) ***MoveJ*** - ruch do punktu docelowego w przestrzeni maszyny (*joint space*).

W naszym programie zastosujemy następujące komendy:

Path_10
MoveJ Target_10
MoveJ Target_30
MoveL Target_20
MoveJ Target_30
MoveJ Target_10

Komenda ***MoveL*** będzie wykorzystana tylko do przejazdu od punktu ***Target_30*** do powierzchni części na stole i z powrotem, pozostałe ruchy będą realizowane przez ***MoveJ***.
Aby zmodyfikować komendy należy:

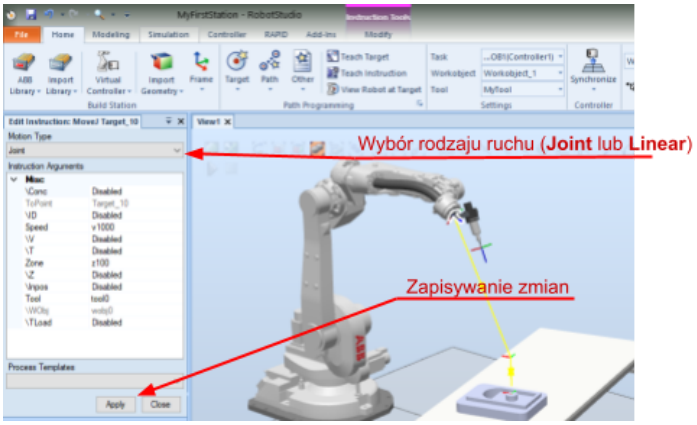
1) kliknąć prawym klawiszem myszy w pierwszą komendę ścieżki i wybrać **Edit Instruction**
2) w górnej części okna, które się otworzy, wybrać z listy odpowiednią wersję komendy ruchu: **Joint** - dla komendy ***MoveJ***, lub **Linear** - dla ***MoveL*** (rys. 15),
3) wcisnąć **Apply**,
4) powtórzyć tę operację dla pozostałych komend w ścieżce ***Path_10***.

Dla bezbłędnego przejścia ścieżki istotne jest też ułożenie robota (*robot configuration* - wzajemny układ osi robota) w kolejnych jej punktach. Aby ruch przebiegał gładko, ułożenie robota w następujących po sobie punktach ścieżki nie powinno się znacząco zmieniać.

Konfigurację robota w języku RAPID zapisuje się w postaci czwórki liczb:

(cf1, cf4, cf6, cfx)

gdzie: **cf1, cf4, cf6** - kwadrant w którym znajduje się oś 1, 4 i 6 (pozycja kątowa osi - tabela 2), cfx - liczba oznaczająca konfigurację robota (tabela 3, rys. 16).



Rys. 15. Okno edycji instrukcji ruchu

Konfigurację robota w każdym z punktów *Target_xx* można sprawdzić i zmienić klikając prawym klawiszem na nazwę punktu na liście ***Paths&Targets*** i wybierając **Configurations**.
W RobotStudio jest narzędzie pozwalające na porównanie konfiguracji robota w kolejnych punktach ścieżki (np. ***Path_10***) - należy kliknąć prawym klawiszem na komendę ścieżki związaną z ruchem (np. MoveJ lub MoveL) i wybrać **Modify Instruction → Configurations…** (rys. 16).

Zadanie 4: Korzystając z opisu powyżej należy ułożyć robota w następujący sposób:

a) ***Target_10*** - (0, 0, 0, 0),
b) ***Target_30*** - (0, 0, -1, 0),
c) ***Target_20*** - (0, 0, -1, 0).

Zapisać położenie przegubów w każdym z punktów.

Tabela 2: Wartości parametrów cf1, cf4 i cf6

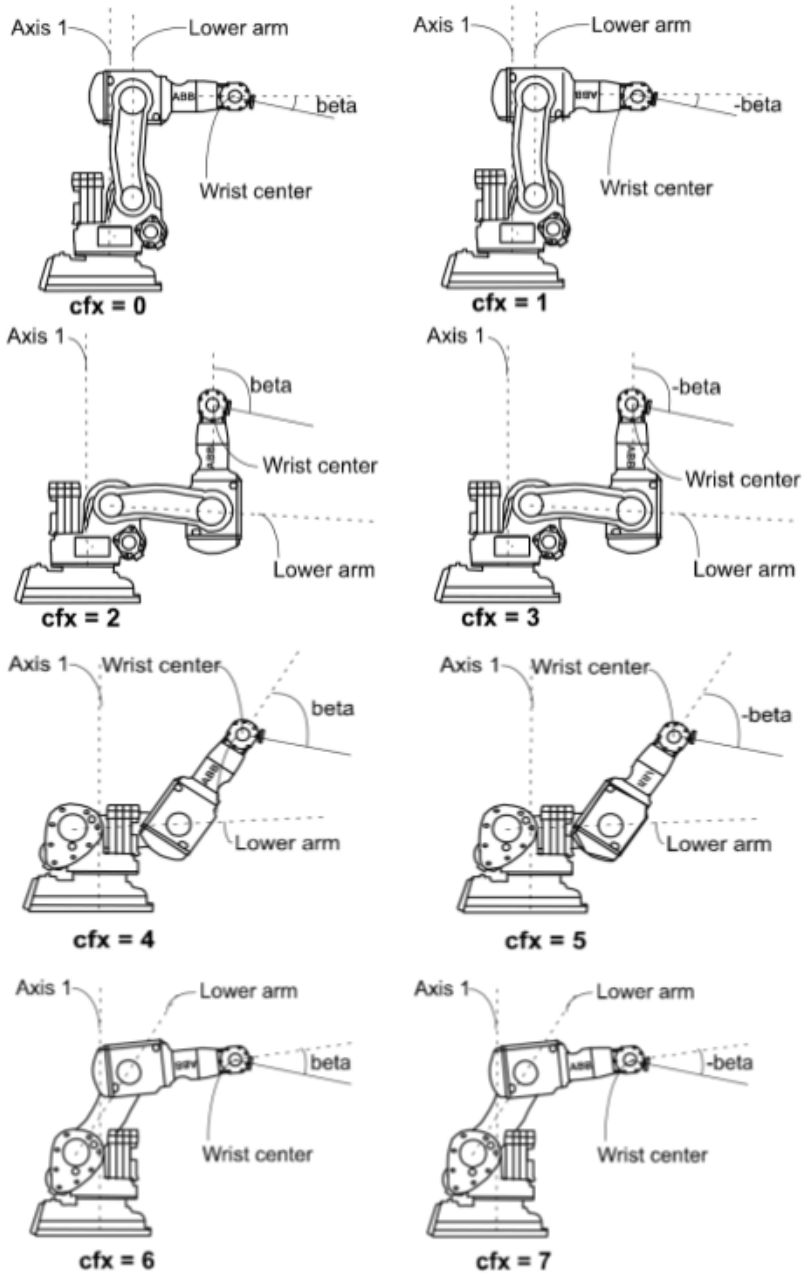
Pozycja przegubu	-360..-270	-270..-180	-180..-90	-90..0	0..90	90..180	180..270	270.360
Wartość parametru	**-4**	**-3**	**-2**	**-1**	**0**	**1**	**2**	**3**

Tabela 3: Wartości parametru cfx określającego konfigurację robota

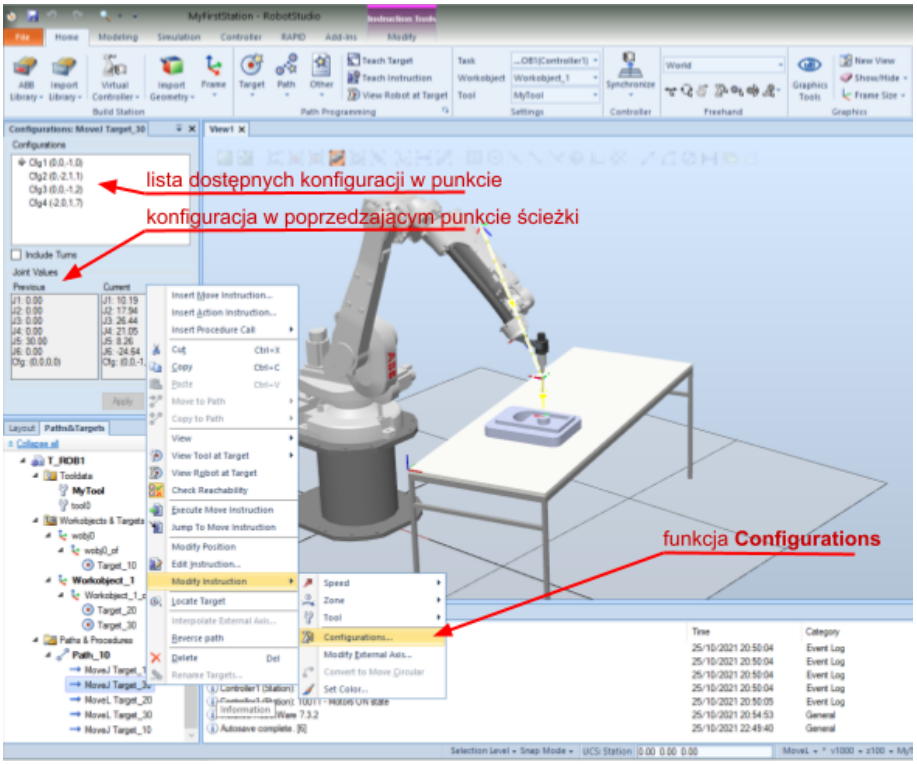
cfx	**położenie nadgarstka w stosunku do osi 1**	**położenie nadgarstka w stosunku do niższego ramienia**	**kąt osi 5**
0	przed	przed	dodatni
1	przed	przed	ujemny
2	przed	za	dodatni
3	przed	za	ujemny
4	za	przed	dodatni
5	za	przed	ujemny
6	za	za	dodatni
7	za	za	ujemny

5.4.1 Weryfikacja poprawności ścieżki

Aby sprawdzić poprawność ścieżki można użyć funkcji **Move Along Path** dostępnej po kliknięciu prawym klawiszem myszy na nazwę ścieżki (w naszym przypadku ***Path_10***). Po uruchomieniu funkcji robot podejmie próbę przejazdu po zaprogramowanej ścieżce. Jeżeli w czasie przejazdu wystąpią błędy, robot zatrzyma się a informacja o błędzie zostanie zapisana do dziennika (logu) pod oknem projektu. Jeżeli robot został ułożony tak jak w zadaniu 4 powyżej, to przejazd po ścieżce powinien odbyć się bez błędów, może jedynie wystąpić ostrzeżenie *Corner Path Failure*, które oznacza, że na jednym z odcinków ścieżki robot ma za mało miejsca na zmianę kierunku do następnego punktu lub niewłaściwie został zdefiniowany punkt końcowy ścieżki.



Rys. 16. Wartości parametru cfx dla różnych ułożeń robota szeregowego (dokumentacja ABB)



Rys. 17. Funkcja Configurations pozwala porównać ułożenie robota w kolejnych punktach ścieżki

Aby usunąć problem z ***Corner Path Failure*** należy:

1) kliknąć prawym klawiszem myszy na ostatnią komendę ścieżki (***MoveJ Target_10***) i wybrać kolejno **Modify Instruction → Zone → fine**,
2) dodatkowo tę samą operację można wykonać dla instrukcji dojazdu do powierzchni części (***MoveJ Target_20***) - wtedy narzędzie robota dojedzie dokładnie do punktu docelowego i się w nim zatrzyma.

5.5 Uruchomienie programu w sterowniku robota

Sprawdzoną ścieżkę wraz ze zdefiniowanymi punktami zapiszemy do sterownika robota. Po tej operacji można będzie przeprowadzić symulację pracy robota. Należy:

1) prawym klawiszem myszy kliknąć w nazwę ścieżki **Path_10** i wybrać **Set as simulation entry point** - koło nazwy ścieżki pojawi się informacja **(entry point)**,

2) na liście ***Paths&Targets*** kliknąć prawym klawiszem myszy na sterownik **Controller1** i wybrać z menu funkcję **Synchronize to RAPID**,
3) sprawdzić, czy w otwartym oknie zaznaczone są wszystkie pozycje w kolumnie **Synchronize** i kliknąć **OK** - program zostanie zapisany do sterownika co zostanie też odnotowane w dzienniku zdarzeń pod oknem projektu.

Program jest gotowy do uruchomienia. Teraz można uruchomić symulację:

4) zmienić zakładkę w głównym pasku programu z **Home** na **Simulation**,
5) kliknąć **Play** - symulacja zostanie uruchomiona,

Aby ukryć przybliżony zarys ścieżki i włączyć rejestrowanie trajektorii narzędzia należy:

6) kliknąć prawym klawiszem myszy na **Path_10**, wybrać **View → Visible**,
7) kliknąć na ikonę **TCP Trace** w pasku ***Simulation*** - w otwartym oknie zaznaczyć **Enable TCP Trace**, i ewentualnie wybrać kolor klikając na barwne pole **Primary Color**,
8) uruchomić symulację (przyciskiem **Play**).

Trajektoria punktu TCP będzie składała się z dwóch odcinków - jeden dla narzędzia ***tool0*** (w okolicy punktu ***Target_10***), drugi dla narzędzia ***MyTool***. Jest to konsekwencją przyjętych wcześniej założeń (*Target_10* został przypisany do narzędzia *tool0*).

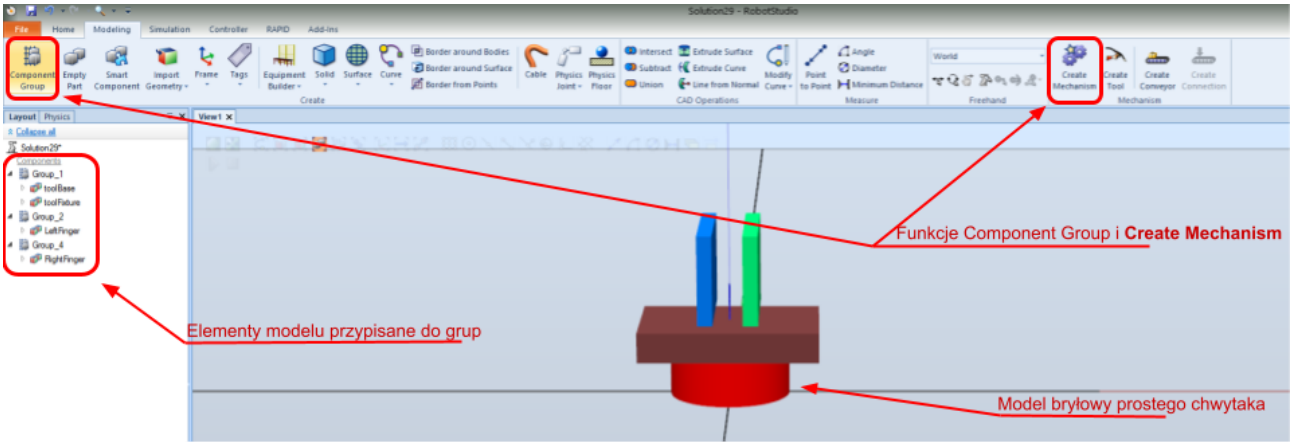
6 Tworzenie i testowanie własnych narzędzi robota

Z modelu bryłowego (np. chwytaka lub zaawansowanej głowicy robota) zaimportowanego do RobotStudio można utworzyć narzędzie, zamocować je na robocie i przetestować jego funkcjonalność.

6.1 Tworzenie mechanizmu

Aby utworzyć mechanizm należy najpierw przypisać wszystkie elementy modelu bryłowego do grup w taki sposób aby każda grupa zawierała elementy, które nie przemieszczają się względem siebie (np. baza narzędzia w jednej grupie, lewa łapa chwytaka w drugiej, prawa łapa w trzeciej). Jeżeli elementy zaimportowanego modelu nie są odpowiednio pogrupowane, można do tego użyć modułu **Component Group** z menu **Modelling** (rys. 18).

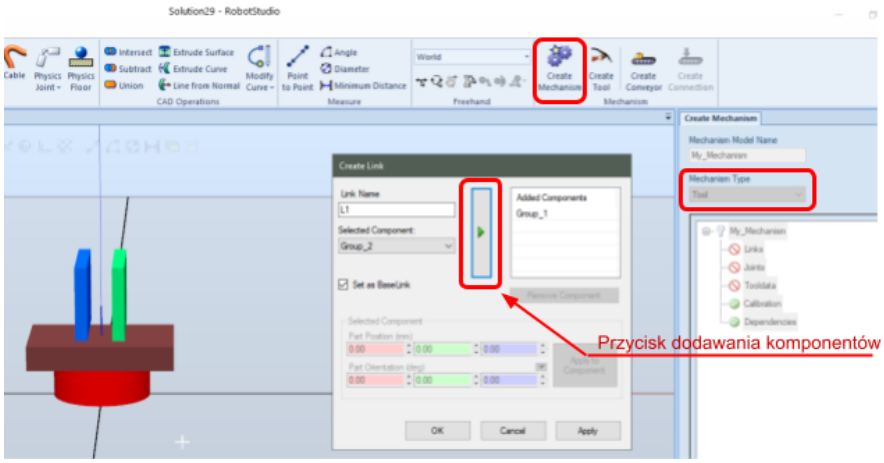
Uwaga: Zaimportowany model narzędzia powinien być tak ustawiony, że jego układ odniesienia związany z bazą narzędzia pokrywał się z głównym układem odniesienia w projekcie RobotStudio (***World***) - pozwoli to później na bezproblemowe zamontowanie narzędzia na robocie.



Rys. 18. Elementy modelu bryłowego chwytaka przypisane do grup funkcjonalnych: Group_1 (elementy podstawy - czerwony i brązowy), Group_2 (lewa łapa - niebieski) Group_4 (prawa łapa - zielony)

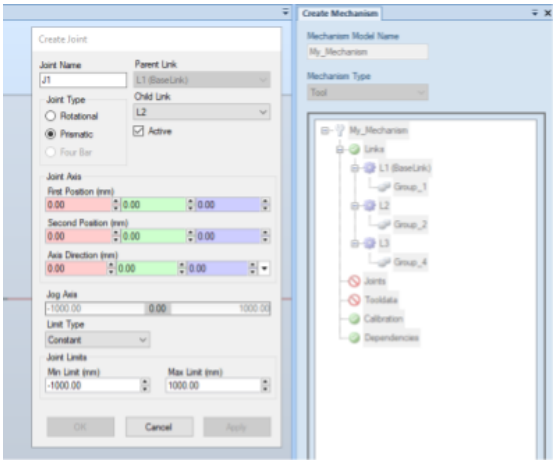
Po utworzeniu grup można utworzyć narzędzie. Należy:

1) wcisnąć przycisk **Create Mechanism** (rys. 18),
2) w oknie ***Create Mechanism*** wybrać rodzaj mechanizmu (*Mechanism Type*) **Tool** - pojawi się lista zawierająca trzy zaznaczone na czerwono elementy: *Links, Joints* i *Tooldata*,
3) kliknąć dwa razy w element **Links** - pojawi się okno wyboru komponentów (**Create Links**; rys. 19), w którym należy zdefiniować moduły, które później zostaną powiązane przegubami,
4) w oknie ***Create Links*** dla modułu **L1** (nazwa ustawiona automatycznie, można ją zmienić) wybrać **Group_1** (podstawa narzędzia) z listy **Selected Component**, zaznaczyć **Set as Base Link** i wcisnąć przycisk dodawania komponentów - okno powinno wyglądać dokładnie jak na rysunku 18,
5) wcisnąć **Apply** - moduł zostanie zapisany w strukturze tworzonego mechanizmu,
6) z listy **Selected Component** wybrać **Group_2**, wcisnąć przycisk dodawania komponentów i potem **Apply**,
7) z listy **Selected Component** wybrać **Group_3**, wcisnąć przycisk dodawania komponentów i potem **OK**.



Rys. 19. Funkcja Create Link służy do zdefiniowania części mechanizmu, które później zostaną powiązane przegubami

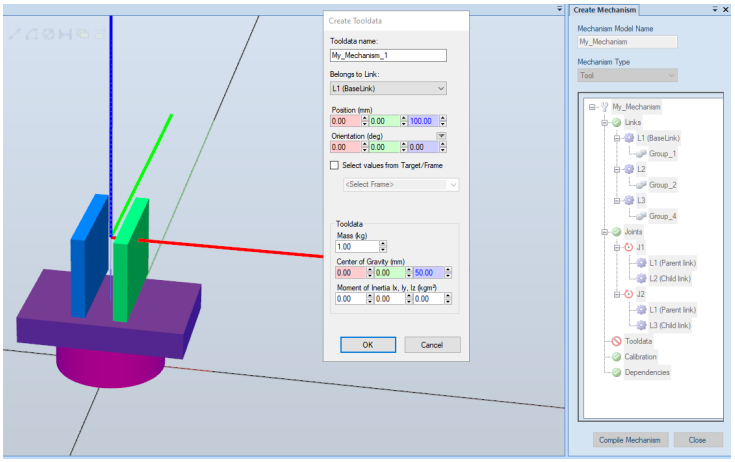
8) Po zdefiniowaniu modułów narzędzia (*Links*) można zdefiniować powiązania między nimi (**Joints**) - należy kliknąć dwa razy w **Joints** - otworzy się okno ***Create Joint*** (rys. 20),



Rys. 20. Okno definiowania przegubów

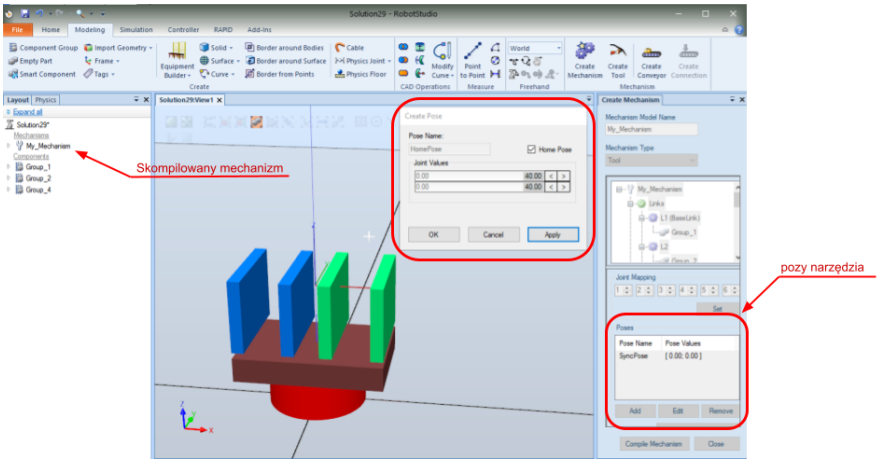
9) korzystając z dostępnych opcji należy zdefiniować kolejno wszystkie przeguby w mechanizmie, określając ich typ (**Rotational** - obrotowy, lub **Prismatic** - przesuwny), moduł bazowy (**Parent Link)** i moduł ruchomy (**Child Link**) oraz określając kierunek osi obrotu przegubu (**Axis Direction)** lub kierunek przesuwu przegubu liniowego (**First Position** i **Second Position**),

10) określić zakres ruchu przegubu (**Joint Limits**) i sprawdzić, czy przegub pracuje prawidłowo korzystając z linijki **Jog Axis**,
11) po sprawdzeniu przegubu wcisnąć **Apply** i wykonać te same operacje dla pozostałych przegubów (UWAGA: w przypadku chwytaka z rys. 17 oba przeguby będą zaczepione do tej samej bazy - dla obu przegubów należy ustawić **Group_1** jako **Parent Link**),
12) ostatnim krokiem koniecznym do zdefiniowania mechanizmu jest ustalenie punktu centralnego narzędzia (TCP) oraz określenie położenia środka ciężkości i momentu bezwładności w oknie **Create Tooldata** (rys. 21)



*Rys. 21. Okno **Create Tooldata** pozwala zdefiniować punkt centralny narzędzia oraz położenie środka ciężkości i moment inercji potrzebne do obliczeń dynamiki*

13) zdefiniowany mechanizm należy skompilować wciskając przycisk **Compile Mechanism** w dolnej części okna ***Create Mechamism*** który uaktywni się po wpisaniu wszystkich wymaganych parametrów - skompilowany mechanizm pojawi się w liście **Layout** w oknie po lewej stronie ekranu,
14) aby można było przeprowadzić symulację ruchu narzędzia można zdefiniować wybrane pozy (rys. 22) - nie jest to obowiązkowe ale może ułatwić przygotowanie symulacji całej celi - pozy definiuje się używając formularza w oknie **Create Tool**, Po zdefiniowaniu póz należy wcisnąć przycisk **Close** - narzędzie zostanie ustawione w jednej ze zdefiniowanych póz a z listy ***Layout*** usunięte zostaną grupy zawierające modele części mechanizmu - pozostanie jedynie utworzony mechanizm.



Rys. 22. Definiowanie póz narzędzia - położenie elementów ruchomych w pozie określa się ustawiając wartości w Joint Values. Na ekranie widoczne jest narzędzie w pozie początkowej (łapy razem) i w definiowanej pozie Home (łapy rozsunięte).

15) Okno modyfikacji narzędzia można otworzyć ponownie klikając prawym klawiszem na nazwę narzędzia w liście **Layout** i wybierając **Modify Mechanism …**

Uwaga: Dla uzyskania symulacji odzwierciedlającej warunki rzeczywiste należy ustawić czas ruchu narzędzia przy zmianie pozy - służy do tego funkcja **Set Transition Times** dostępna pod przyciskiem w dolnej części okna **Create Mechanism**.

6.2 Montaż narzędzia na robocie

Aby zamontować narzędzie na robocie wystarczy przeciągnąć skompilowany mechanizm (element z listy ***Layout***) i upuścić go na element z nazwą robota na liście ***Layout***. Procedura jest identyczna z opisaną w rozdziale ***Tworzenie modelu celi robota*** (rozdział 5.1).

7 Rozwiązywanie problemów z RobotStudio

7.1 Program nie działa - co robić?

W tym rozdziale zebrane są podpowiedzi jak rozwiązać najczęściej występujące problemy z RobotStudio.

#1 Jedna (lub więcej) funkcji w programie przestaje działać chociaż wcześniej nie było z nią problemu

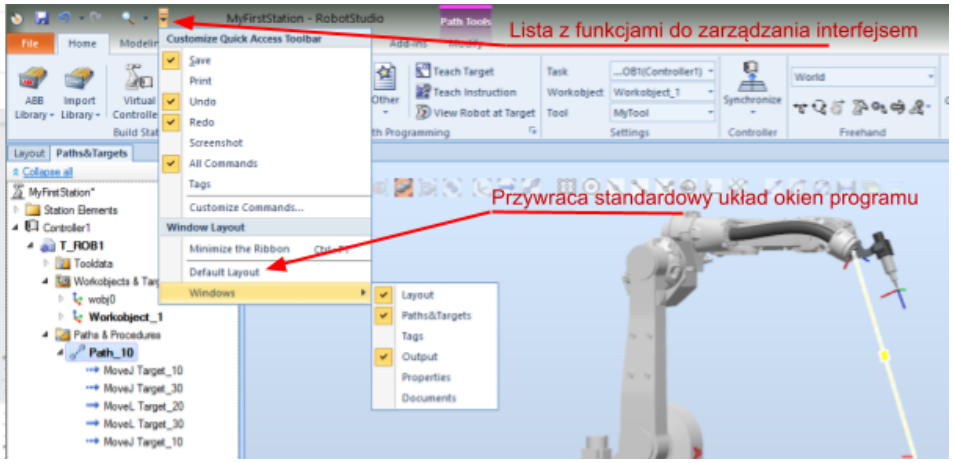
Należy zapisać projekt i zrestartować program - po ponownym otwarciu projektu (uwaga - należy zaczekać aż sterownik robota będzie gotowy do pracy) zwykle problem znika.
Przy powtarzających się problemach najlepiej pobrać i zainstalować najnowszą wersję RobotStudio

#2 Niektóre funkcje menu przestały być aktywne (nie działają)

Proszę sprawdzić ważność licencji lub połączenie z serwerem licencji. Program RobotStudio można używać bez ważnej licencji ale jego funkcje są wtedy ograniczone.

#3 Nie widzę jednego z okien programu (np. listy Layout)

W górnym pasku okna programu jest rozwijana lista z funkcją resetu interfejsu (wszystkie elementy interfejsu przywracane są do układu standardowego) lub włączenia poszczególnych okien (rys. 23)



Rys. 23. Przywracanie standardowego wyglądu interfejsu programu

7.2 Procedura instalacji licencji RobotStudio

Licencję na program RobotStudio można "wypożyczyć" z serwera - pozwala to na korzystanie z programu bez aktywnego połączenia z siecią lokalną w poza laboratorium. W RobotStudio licencja tego typu nazywa się ***commuter license***.
Commuter license działa przez określony czas, np. 90 dni, w każdej chwili można ją zwrócić na serwer. Po upływie terminu ważności i po uzgodnieniu z opiekunem laboratorium można "wypożyczyć" nową licencję.

Aby uzyskać licencję typu commuter license należy:

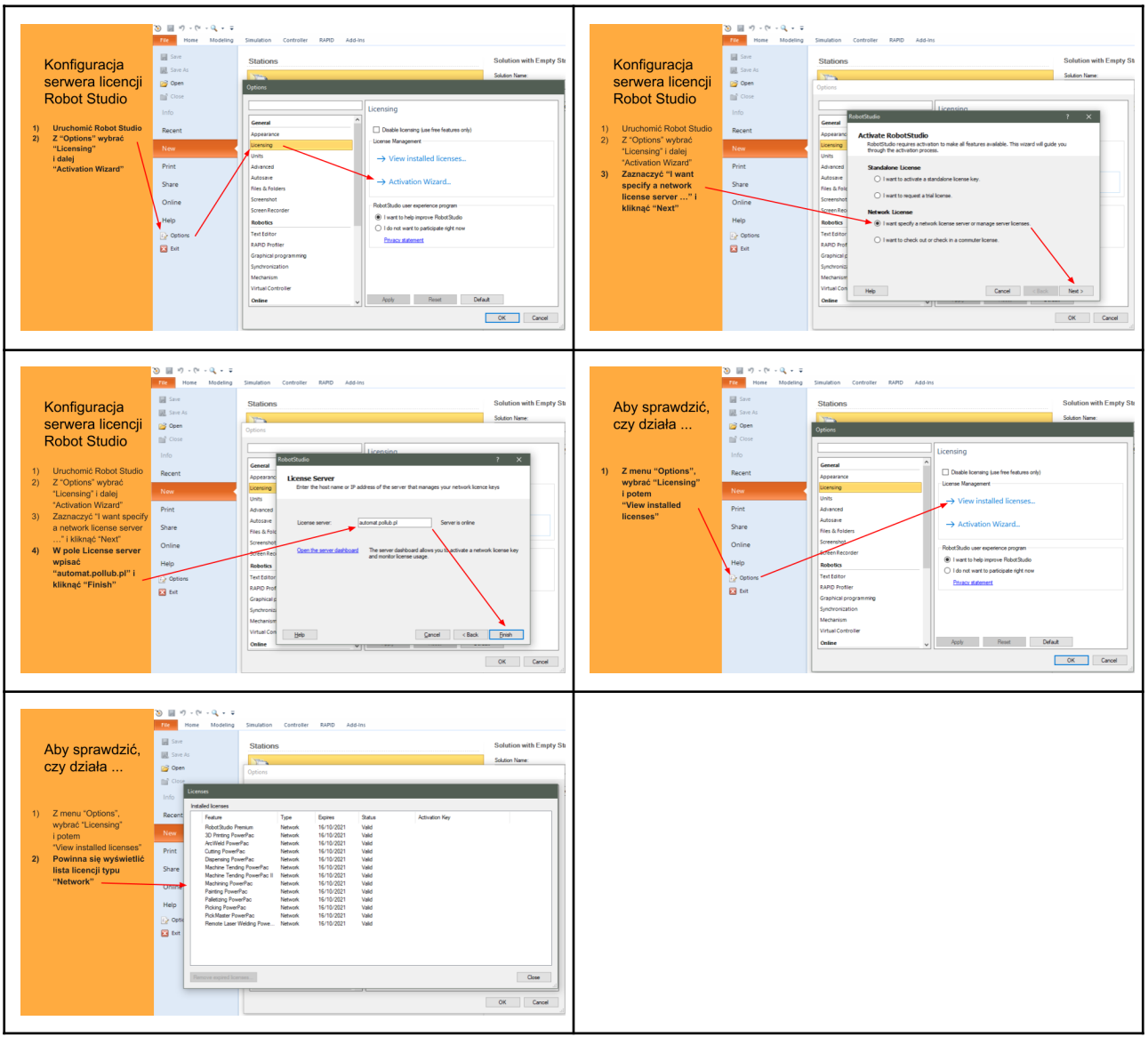
- skonfigurować połączenie do serwera licencji
- wypożyczyć licencję

Uwaga: Fakt wypożyczenia licencji należy potwierdzić wysyłając e-mail zawierający:

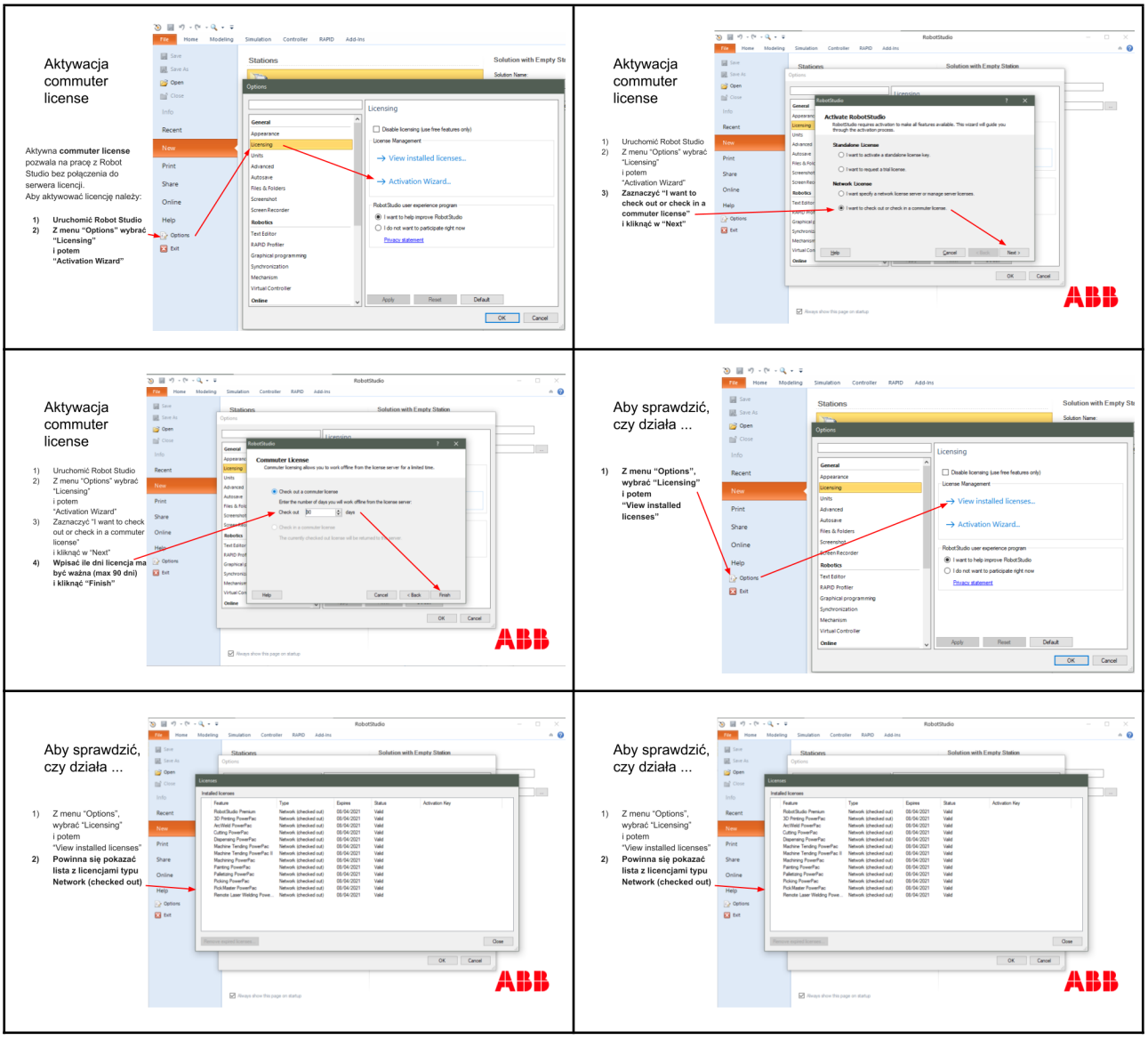
1. imię i nazwisko,
2. nazwę komputera (zapisaną w ustawieniach systemu Windows),
3. datę ważności licencji.

Wiadomość e-mail należy wysłać na adres opiekuna Laboratorium Automatyki Przemysłowej (r.cechowicz@pollub.pl)

7.2.1 Procedura konfiguracji połączenia do serwera licencji



7.2.2 Procedura wypożyczenia licencji z serwera



Ostatni krok

1) Z menu "Options", wybrać "Licensing" i potem "View installed licenses"
2) Powinna się pokazać lista z licencjami typu Network (checked out)
3) **Sprawdzić nazwę komputera w ustawieniach Windows i wysłać e-mail (opis na pierwszym slajdzie)**