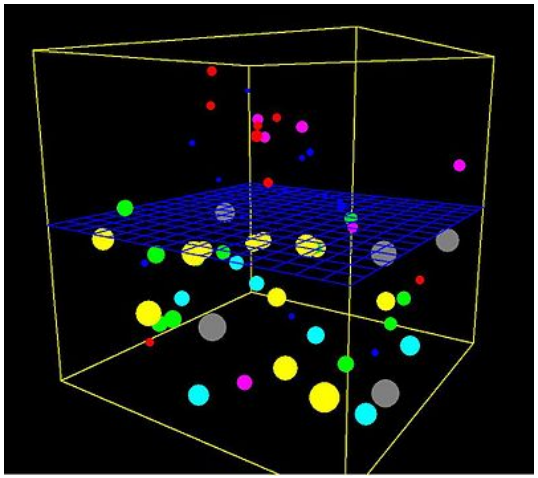
AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Środowiska Symulacji Robotów

12.06.2013 Kraków

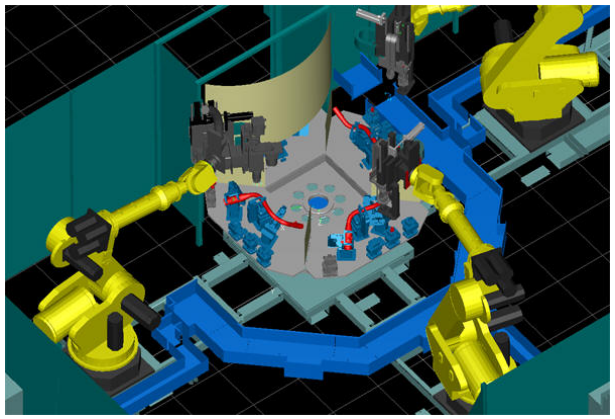
Symulacja komputerowa

- **badanie** własności systemu przy pomocy komputera
- oparte na matematycznym **modelu** rzeczywistości
- model odwzorowujący **niektóre** aspekty rzeczywistości
- model:
  - niedokładny,
  - wyidealizowany
- przetwarzany numerycznie



Symulacje

- rosnąca rola symulacji we współczesnym przemyśle



- gałęzie przemysłu stosujące symulacje
  - przemysł maszynowy,
  - przemysł samochodowy,
  - przemysł lotniczy

- korzyści wynikające ze stosowania symulacji:
  - mniejsze koszty opracowania rozwiązań,
  - krótszy termin wprowadzenia produktu do użytku

- etapy projektowania robota wykorzystujące symulacje
  - projektowanie mechaniczne,
  - projektowanie elektryczne,
  - projektowanie informatyczne

Do czego służy symulator robota?

Symulator komputerowy umożliwia odtwarzanie rzeczywistego przebiegu pracy robota, wykorzystując modele matematyczne.

Programy symulacyjne są wykorzystywane do szkolenia nowych pracowników, zanim przystąpią do pracy na rzeczywistym robocie.

Możliwość poznania i testów elementów:

- programowania,
- sterowania,
- obsługi

symulowanej jednostki roboczej.

Firmy produkujące roboty udostępniają podstawowe narzędzia programistyczne i symulacyjne.

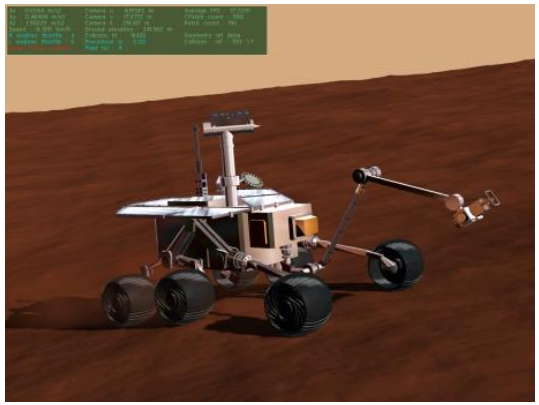
Główne cele symulacji robota i otoczenia

- szybsze testowanie algorytmów zachowania robota w różnych sytuacjach,

- symulowanie zachowań niedostępnych w rzeczywistym otoczeniu,

- zmniejszenie kosztów testowania i sprzętu,

- zmniejszenie ryzyka uszkodzenia sprzętu w wyniku nieprawidłowego zachowania robota podczas pierwszych faz uruchomienia

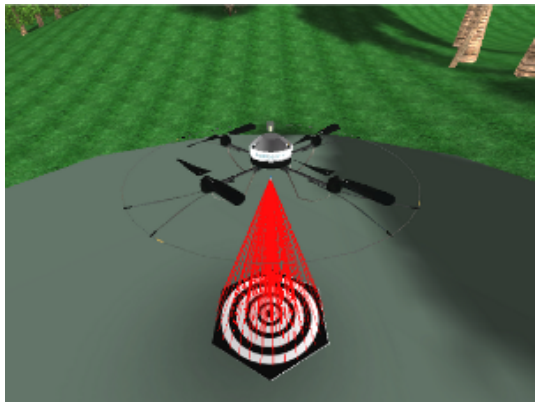


- wykluczenie błędów algorytmów,

- dostarczenie danych dla usprawnienia konstrukcji mechanicznej i elektronicznej robota.

Dlaczego używamy symulatorów?

- koszt urządzeń- możliwość przeprowadzenia symulacji bez konieczności posiadania robota
- koszt czasowy
  - opracowanie nowego robota to min 2 lata
  - testy przydatności robota do realizacji zadania
- uzyskane wyniki symulacji są prawdziwe dla zastosowanego modelu
- zgodność z rzeczywistością jest uzależniona od właściwości modelu

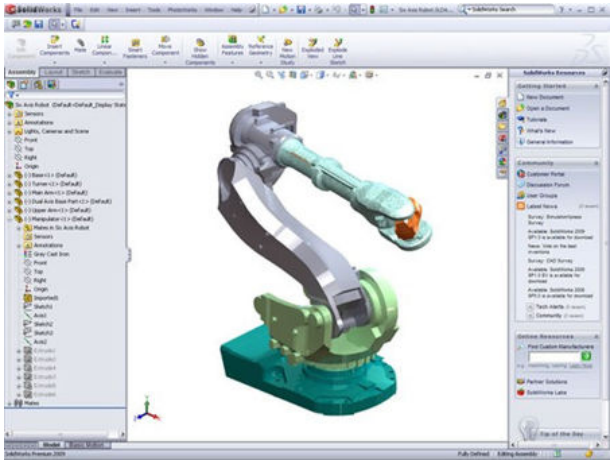


Zalety:

- testowanie właściwości projektowanych konstrukcji (prototypów)
- możliwość modelowania urządzeń których fizycznie nie da się skonstruować
- możliwość powtarzania eksperymentów
- bezpieczeństwo ludzi i sprzętu

Uniwersalne środowisko symulacyjne robota

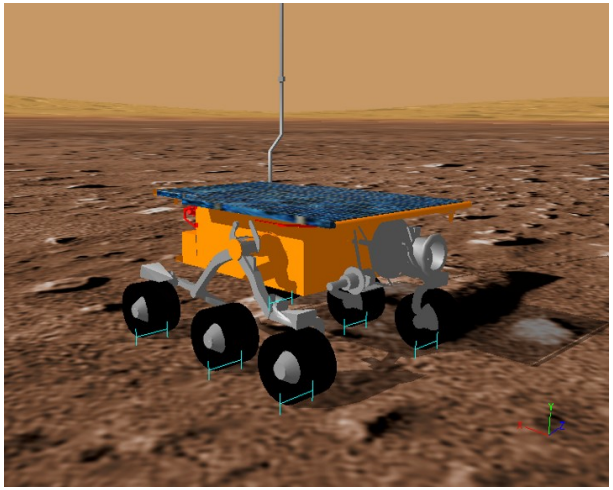
- Modelowanie i symulowanie obiektów 3D (układów połączonych brył)
- Możliwość reprezentowania złożonych urządzeń mechanicznych (stopnie swobody, przykładanie sił)
- Realistyczna symulacja kinematyki i dynamiki brył sztywnych
- Wparcie konstruowania typowych elementów robotów
- Wykrywanie kolizji
- Uruchamianie programów symulacyjnych w rzeczywistości
- Wydajność
- Możliwie wiele języków i technologii
- Sterowanie robotami, symulacją
- Programowanie wizualne
- API



Kryteria doboru środowiska symulacji:

Symulowanie zachowania **robota mobilnego w otoczeniu** wymaga możliwości:

- Tworzenia trójwymiarowych modeli otoczenia
- Symulowania działania modelu w czasie rzeczywistym
- Symulowania interakcji robota z otoczeniem
- Symulowania pracy czujników:
  - ultradźwiękowych,
  - skanerów laserowych 2D i 3D,
  - GPS,
  - żyrokompasu,
  - akcelerometrów,
  - kamer

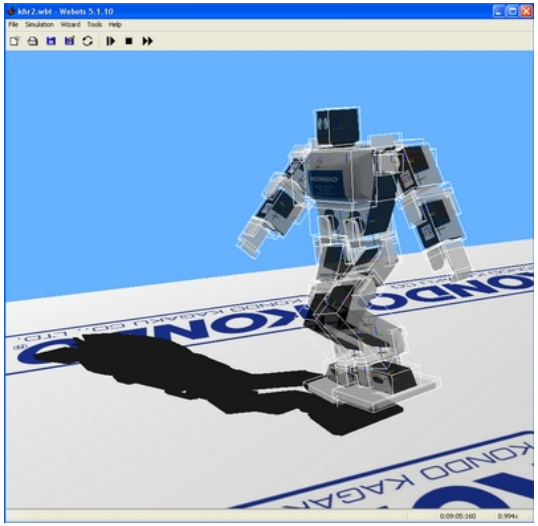


Narzędzia wspomagające symulatory

Dodatkowe narzędzia i moduły rozszerzające funkcjonalności symulatorów.

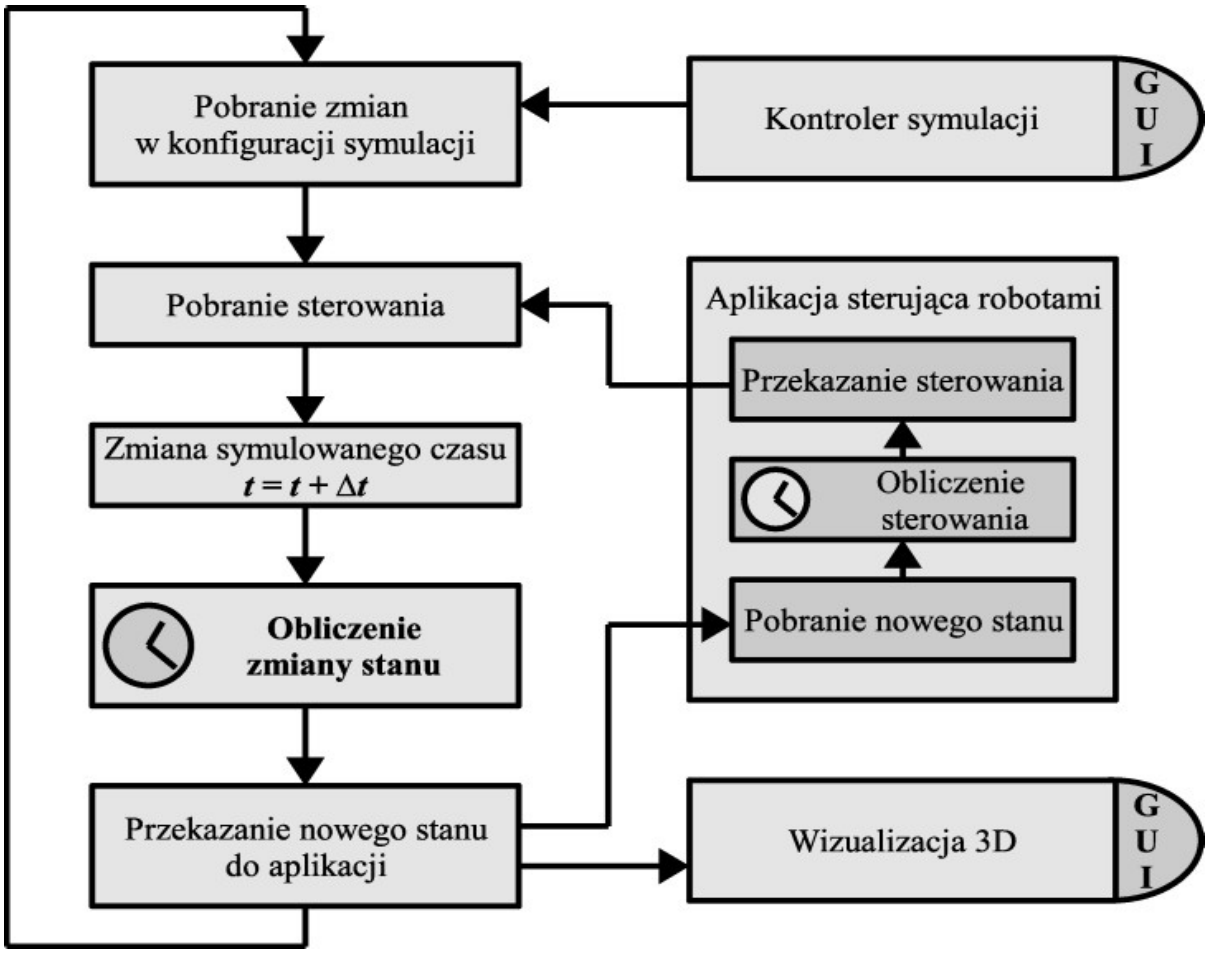
Kategorie rozszerzeń:

- zapisywanie i odtwarzanie przebiegu symulacji
- wizualizacja symulacji
- analiza wyników



ANATOMIA SYMULATORA ROBOTÓW

Cykl symulacji



Krótki opis dostępnych symulatorów robotów

Aerosim Blockset

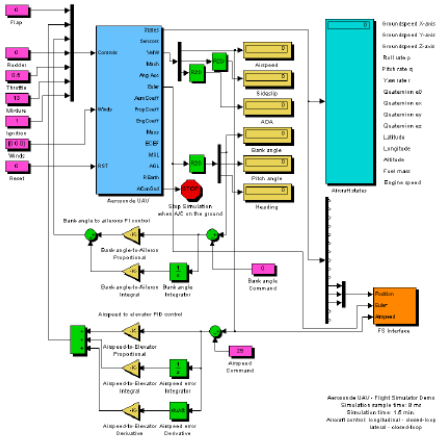
http://www.mathworks.com/products/aeroblks/

- Biblioteka dla programu Matlab/Simulink
- Dostarcza komponenty umożliwiające projektowanie modeli samolotów (6-DOF)
- Posiada narządza symulujące atmosferę, wiatr, turbulencje i modelujące ziemię (geoida, grawitacja, pole magnetyczne)
- Dostępna darmowa wersja akademicka

ARS MAGNA: Abstract Robot Simulator

http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/testbeds/arsmagna/0.html

- abstrakcyjny model świata w którym użytkownik kontroluje robota mobilnego
- napisany w języku LISP dla środowisk linuksowych na licencji GPL.
- Projekt zakończony



Biorobotic Simulation Program

http://www.life.illinois.edu/delcomyn/RSSimSoftware.html

- Język skryptowy o nazwie Config++ oparty o C++
- używany do symulacji dynamiki wielosegmentowych robotów.
- Licencja tylko dla użytkowników niekomercyjnych.

Krótki opis symulatorów - cd.

BugWorks

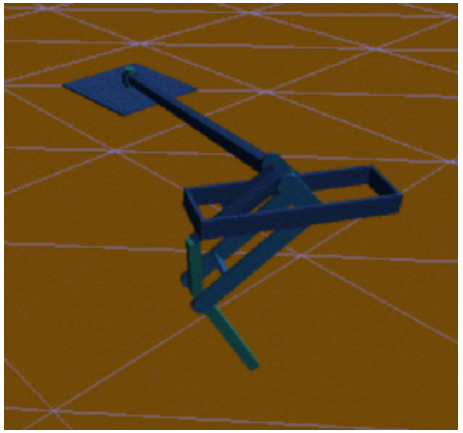
http://www.bugworks.org/

- Dwuwymiarowa symulacja robota w edytorze wizualnym
- Możliwość wbudowania w robota czujników zbliżeniowych
- Dostępny język skryptowy (proste obliczenia numeryczne, instrukcje warunkowe i skoki bezwarunkowe)
- Wersja darmowa pozwala na pełne wykorzystanie możliwości symulacji, bez możliwości zapisywania i ładowania projektów.
- Napisany w JAVA

DYNAMECHS

http://sourceforge.net/projects/dynamechs/

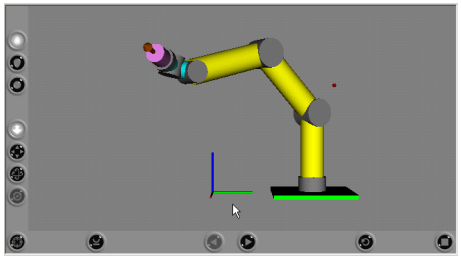
- Biblioteka napisana w C++
- Silnik fizyki i dynamiki brył sztywnych
- Pozwala na generowanie trójwymiarowego otoczenia oraz modelowania robotów.
- Oparta na licencji GNU GPL.



Krótki opis symulatorów - cd.

DynawizXMR

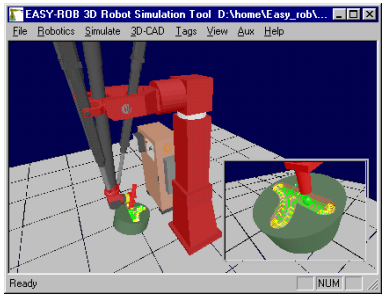
http://www.concurrent-dynamics.com/xmr/



- symulator dynamiki i kontroli mobilnych robotów naziemnych
- napisany w środowisku Matlab/Simulink
- symulacja ciał sztywnych i elastycznych oraz sił działających na elementy
- licencja komercyjna

EASY-ROB

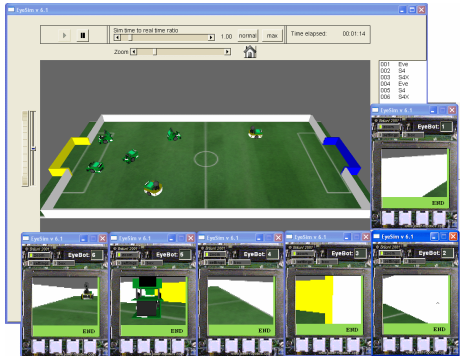
http://www.easy-rob.de/



- Program służący do planowania i modelowania linii robotów przemysłowych.
- Modelowanie sekwencji ruchów, zasięgu, detekcja kolizji, współpraca robotów
- Wizualizacja 3D
- licencja komercyjna, środowisko Windows

EyeSim

http://robotics.ee.uwa.edu.au/eyebot/doc/sim/sim.html



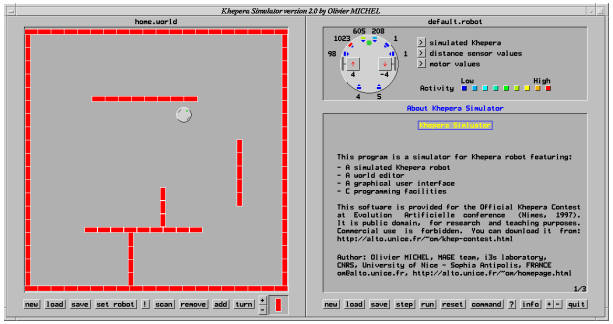
- Symulator robotów mobilnych klasy EyeBot (C++)
- Udostępnia fizykę platformy (kamery pokładowe, czujniki podczerwieni detektory zderzeń, odometria)
- Wizualizacja 3D

Krótki opis symulatorów - cd.

Khepera Simulator

http://diwww.epfl.ch/lami/team/michel/khep-sim/

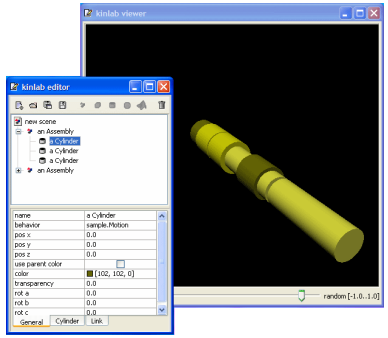
- Dwuwymiarowy symulator robota Khepera
- Algorytmy tworzone w C i C++
- Włączony w skład symulatora WEBOTS



Kinlab

http://kinlab.dyndns.org/

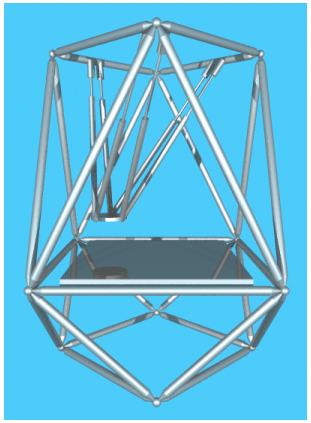
- Modelowanie dynamiki robota przemysłowego
- Sztywne modele kinematyczne z różnorodnymi połączeniami
- Oprogramowanie oparte na Java i Java3D
- Oprogramowanie darmowe



LME Hexapod

http://www.laboratoryformicroenterprise.org/lme/LMEHexapodMachine.html

- Środowisko symulacji robota sześcionożnego.
- Dołączone są źródła w języku C++.
- Licencja GNU GPL.

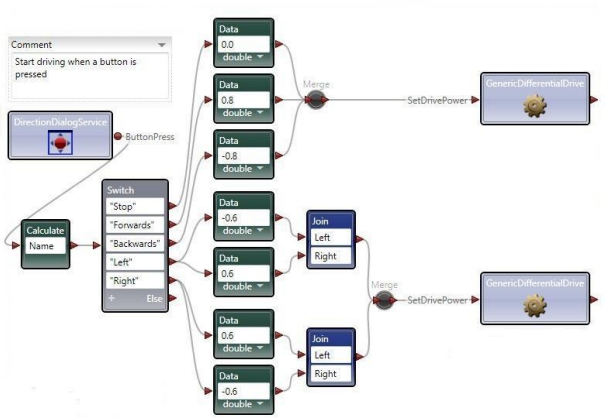


Krótki opis symulatorów - cd.

Microsoft Robotics Developer Studio

http://www.microsoft.com/robotics/

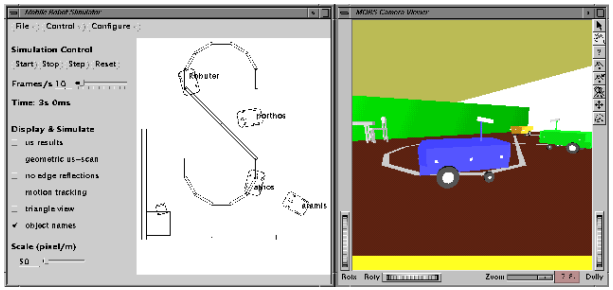
- rozbudowane środowisko symulacji robotów
- oparty na technologii dostarczania usług (REST)
- łączenie się z usługą dostarcza odpowiedniej funkcjonalności, pozwalającej symulować, lub sterować robotem
- nieodpłatna wersja akademicka
- wersja Express bez pełnej funkcjonalności



MOBS – Mobile Robot Simulator

http://robotics.ee.uwa.edu.au/mobs/

- symulator istniejącego robota „Robuter II”.
- modelowanie sygnałów z czujników i widoku z kamery robota.
- możliwość tworzenia własnego trójwymiarowego otoczenia
- napisany w języku C.



OpenDynamicEngine

http://www.ode.org/

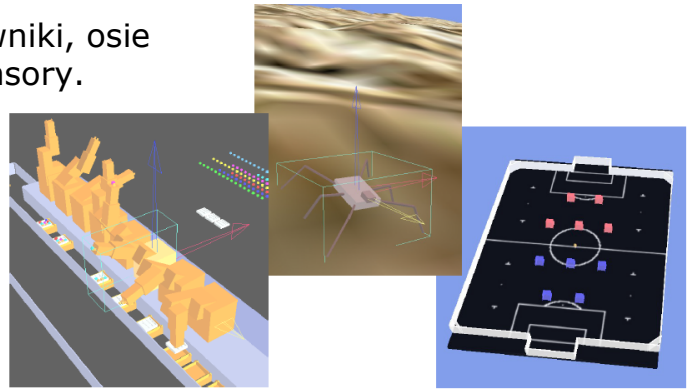
- biblioteka napisana w języku C++
- symulacje dynamiki ciała sztywnego, pojazdów naziemnych, robotów i obiektów ruchomych.
- obsługuje dynamikę ruchu stawów, tarcie i detekcje kolizji.
- oparte na licencji GNU GPL.

Krótki opis symulatorów - cd.

RoBOSS

http://sourceforge.net/projects/roboss/

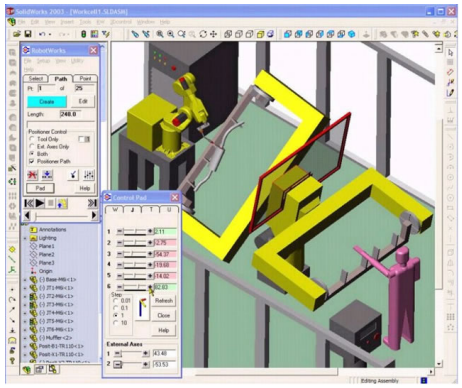
- System do symulacji i testowania konstrukcji oraz algorytmów dla robotów.
- Umożliwia budowanie robotów o dowolnych kształtach i właściwościach.
- Symuluje różne połączenia pomiędzy częściami robotów: siłowniki, osie oraz elementy aktywne jak silniki, serwomechanizmy oraz sensory.
- Możliwość symulacji na wielu węzłach
- Dostępne języki programowania C++ i C#
- Dostępne są źródła oraz skompilowane wersje programu.
- twórcy – W.Turek, D.Czyrnek, AGH, EAIiE, KI, 2005



RobotWorks

http://www.robotworks-eu.com/

- Interaktywne środowisko CAD
- modelowanie i symulacja ruchu robotów przemysłowych
- Generowanie ścieżek i planów na podstawie modeli CAD
- Rozwiązanie komercyjne.

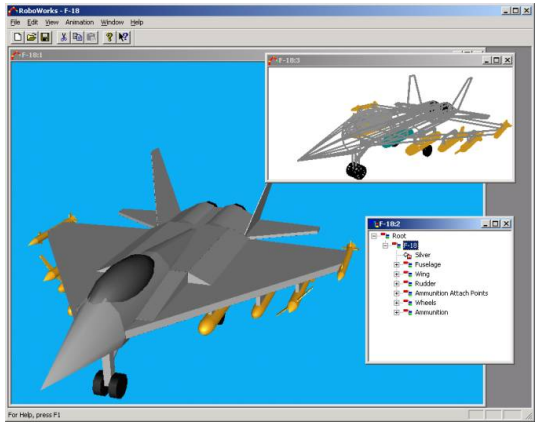


Krótki opis symulatorów - cd.

RoboWorks

http://www.newtonium.com/

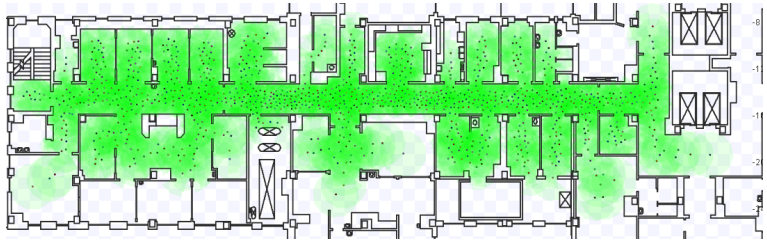
- Narzędzie do modelowania i symulowania systemów fizycznych
- Pozwala na modelowanie dowolnego obiektu (robota)
- Udostępnia interfejs programistyczny do wykorzystania w językach umożliwiających korzystanie z bibliotek dll
- Wersja komercyjna

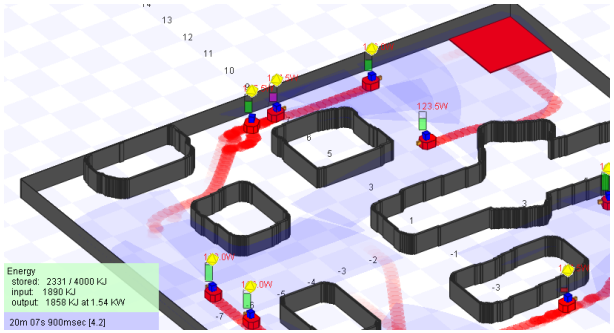


Stage

http://playerstage.sourceforge.net

- Symulator populacji robotów mobilnych w 2D
- Symulacja sensorów i obiektów środowiska
- Cel – wsparcie tworzenie algorytmów działających na wieloagentowych środowiskach autonomicznych robotów
- Dostęp do symulatora z kontrolera przez API
- Dostępny także jako biblioteka C++



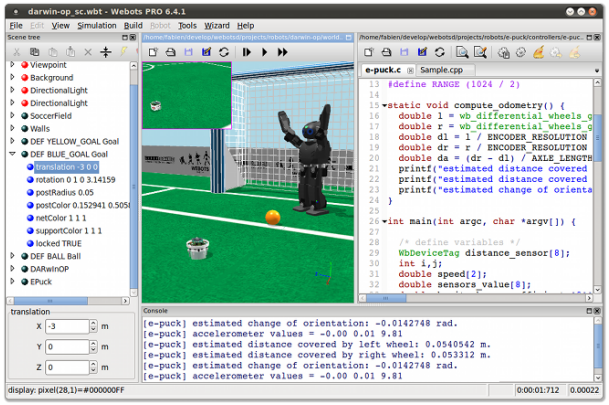


Krótki opis symulatorów - cd.

Webots

http://www.cyberbotics.com/

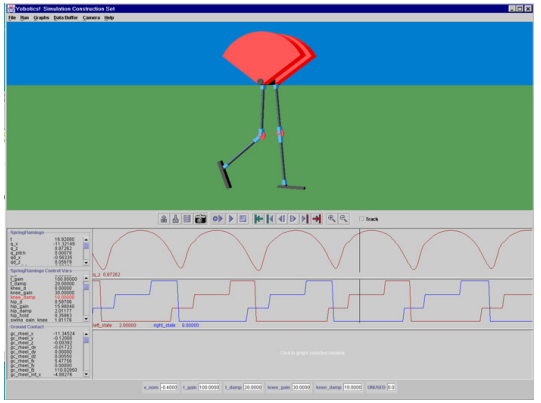
- tworzenie środowisk i robotów 3D
- tworzenie robotów kołowych, nożnych lub latających
- API do kontroli robotów (C,C++, JAVA, Python, Matlab,...)
- pełna fizyką opartą na silniku Open Dynamic Engine
- duża biblioteka modeli czujników, skanerów, kamer i urządzeń komunikacyjnych.
- rozwiązanie komercyjne (Windows, Linux i MAC OS X)



Yobotics! Simulation Construction Set

http://yobotics.com/simulation/simulation.htm

- symulator dynamiki ciała.
- przeznaczony do symulacji robotów kroczących, humanoidów, heksapodów, rąk bądź chwytaków i innych mechanizmów
- monitorowanie parametrów obiektów w czasie rzeczywistym
- darmowy do użytku niekomercyjnego



Porównanie wybranych środowisk symulacji robotów

Nazwa środowiska	Wbudowane wsparcie dla grupowego projektowania	Działanie w środowisku rozproszo-nym	Wsparcie dla istniejących środowisk CAD/CAM	Modelowa-nie otoczenia i interakcji z nim
DYNAMECHS				TAK
eyeSim				TAK
eyeWyre Simulation Studio				TAK
Microsoft Robotics Developer Studio 2008	TAK	TAK	TAK*	TAK
OpenDynamicEngine				TAK
ROBOOP				TAK
RoBOSS		TAK		TAK
RoboWorks				TAK
Webots				TAK

* Wspierany format obj i collada

Porównanie wybranych środowisk symulacji robotów cd.

Nazwa środowiska	Wizualizacja 3D	Kinema-tyka stawów	Dynamika bryły sztywnej	Gotowe biblioteki czujników	Sprzętowe wsparcie obliczeń fizycznych	Darmowe bądź studencka wersja
DYNAMECHS	TAK	TAK	TAK			TAK
eyeSim			TAK	TAK		TAK*
eyeWyre Simulation Studio	TAK	TAK	TAK	TAK		
Microsoft Robotics Developer Studio 2008	TAK	TAK	TAK	TAK	TAK	TAK
OpenDynamicEngine	TAK	TAK	TAK			TAK
ROBOOP	TAK	TAK				
RoBOSS	TAK	TAK	TAK	TAK		**
RoboWorks	TAK	TAK				
Webots	TAK	TAK	TAK	TAK		

* Do rozwiązań niekomercyjnych

** Brak wzmianki o typie licencji

Wybrane symulatory:

- Roboguide
- Roboss
- Cosimir
- Microsoft Robotics Developer Studio

ROBOGUIDE

http://www.fanucrobotics.com/Products/vision-software/ROBOGUIDE-simulation-software.aspx

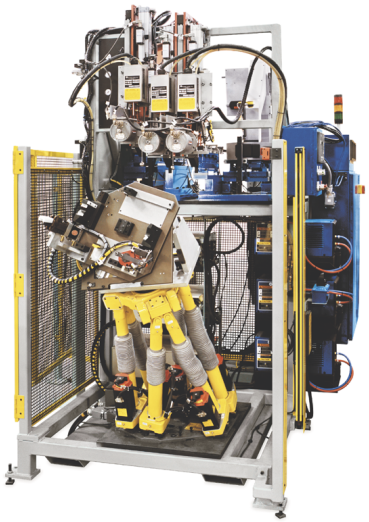
Roboguide

- oprogramowanie symulacyjne, wykorzystujące wirtualny kontroler robota
- symulacja ruchu robota lub wykonalności zadania bez dostępu do rzeczywistych elementów stanowiska pracy.
- weryfikacja operacji robota
  - możliwość wystąpienia kolizji z otoczeniem
  - przebieg operacji wytwarzania przy użyciu animowanej symulacji
  - prezentacja animowanego obrazu robota
- możliwość importu modeli elementów z oprogramowania CAD
- umożliwia:
  - przeprowadzenia symulacji ruchu i wydajności aplikacji zrobotyzowanej
  - oszacowanie czas trwania cyklu pracy
  - wykonać wizualizacji przebiegu procesu
- wirtualny programator (Teach Pendant) symulujący rzeczywiste urządzenie

Roboguide – symulowane roboty

Roboty przemysłowe Fanuc:

- FANUC M-6iB
- FANUC M-10iA
- FANUC F-200iB



Roboty manipulacyjne

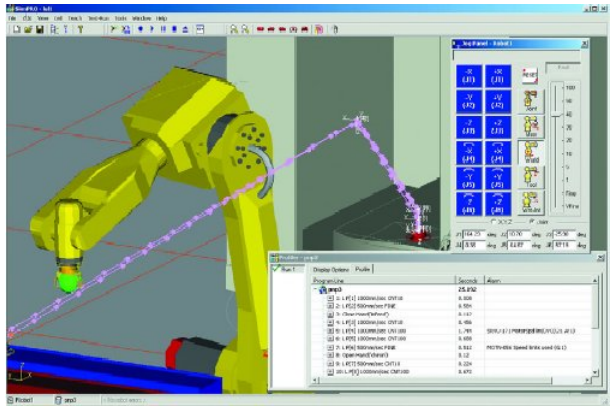
- FANUC M-420iA

Przetwórstwo żywnościowe

- M-421iA

Główne cechy symulatora:

- intuicyjne programowanie offline przy użyciu interfejsu
- przyspiesza integrację systemu
    - redukuje koszty
- wirtualny kontroler osiąga wyniki i czas trwania rzeczywistych cykli
- symulacja działania wszystkich modeli robotów
- wybór i konfiguracja oprogramowania aplikacyjnego:
    - ArcTool (oprogramowanie do spawania łukowego),
    - SpotTool (oprogramowanie do zgrzewania)
    - HandlingTool (oprogramowanie do przenoszenia elementów)

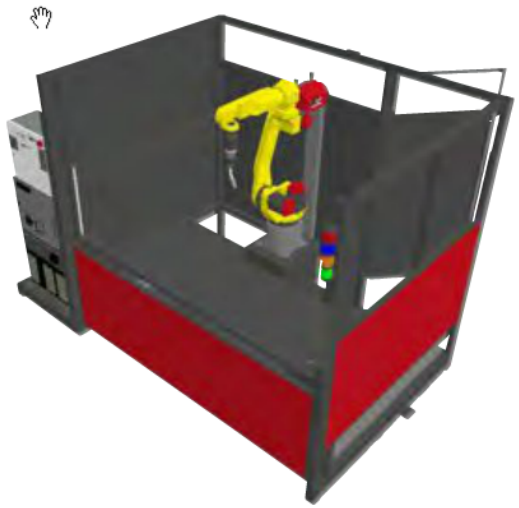


- schemat czasu cyklu, wykrywanie kolizji i obrazy robocze
- ślad ruchu pokazany jako obraz punktowy
- import modeli części roboczych, narzędzi, mocowań i przeszkód w formacie IGES

Roboguide : WeldPro

Weldpro - pakiet dodatkowy dedykowany spawaniu łukowemu

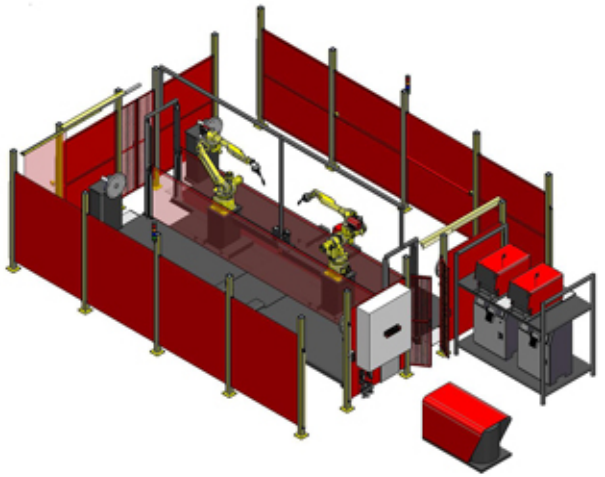
- System spawania konfigurowany przy użyciu modeli elementów 3D CAD
- Możliwość dodania urządzeń peryferyjnych
- Tworzenie programu spawalniczego, w którym operator wyznacza
  - trajektorię,
  - kąt nachylenia palnika spawalniczego,
  - kąt przemieszczenia
- Program spawania weryfikowany przy użyciu animacji.



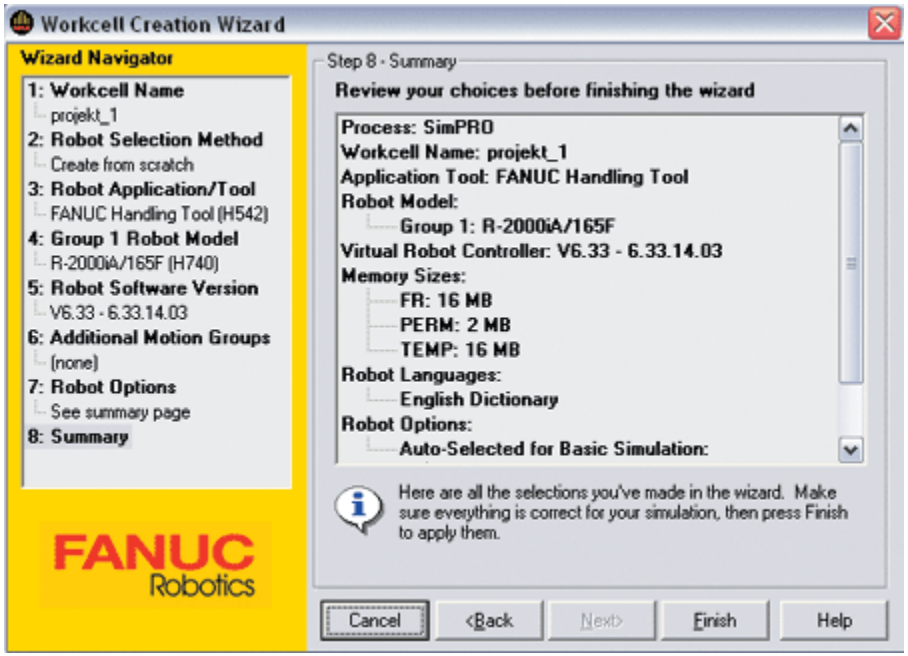
Opcjonalnie: Dual Arm

Dual Arm - opcjonalne oprogramowanie dedykowane aplikacjom spawania łukowego przy użyciu dwóch robotów.

- Robot manipulacyjny podtrzymuje część obrabianą przez robota spawalniczego
- Możliwość wyznaczenia przebiegu spawania na linię modelu CAD
- Automatyczna weryfikacja kolizji pomiędzy obrabianymi częściami i resztą elementów
- Wygenerowany program przesyłany do kontrolera robota.



Przykładowe tworzenie symulacji: robot przenoszący paletę na stół



8 kroków konfiguracji symulacji

krok 1 - nazwa projektu,

krok 2 - projektując wirtualnego robota - tworzymy nowego robota, wykorzystujemy dane z kopii bezpieczeństwa lub tworzymy kopie robota z innego projektu,

krok 3 - wybieramy oprogramowanie, w które wyposażony ma być robot

krok 4 - wybieramy rodzinę i typ robota, dla którego chcemy pisać program,

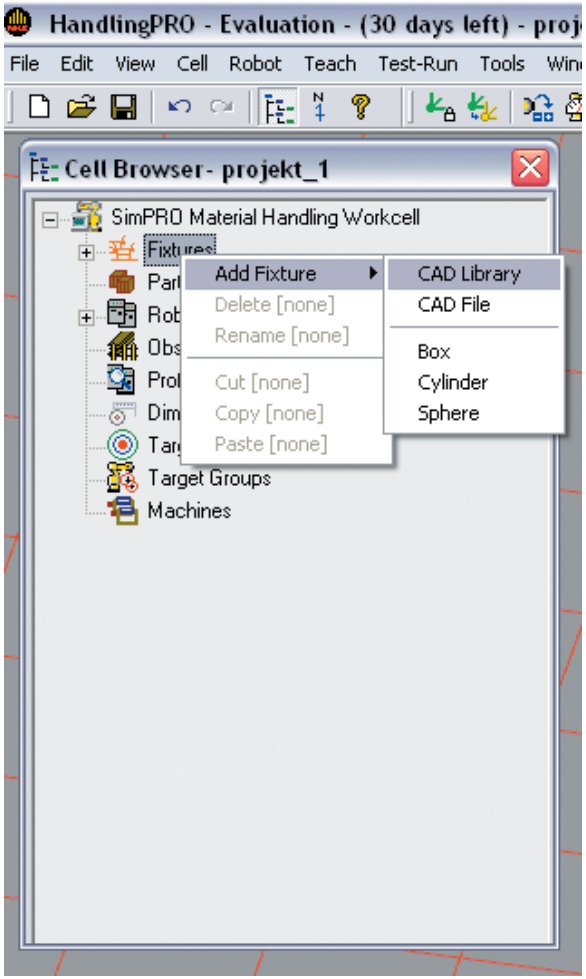
krok 5 - wybieramy wersję oprogramowania, w które wyposażony jest robot

krok 6 - konfiguracja zewnętrznych osi, w które może być wyposażony robot,

krok 7 - opcje robota

krok 8 - podsumowanie

Elementy z otoczenia robota

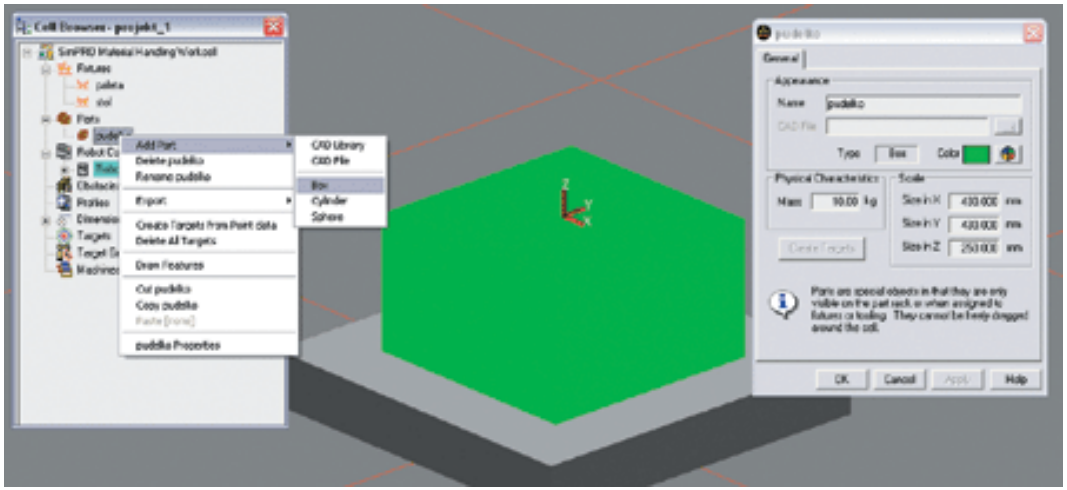


W symulatorze możemy dodać różne elementy otoczenia robota - gotowe elementy dostarczone z oprogramowaniem (CAD Library)

Z biblioteki wybieramy stół, jak i paletę, które spełniają założenia projektu.

Jeżeli wśród gotowych elementów nie możemy znaleźć właściwego, możemy zdefiniować swoje pliki elementów.

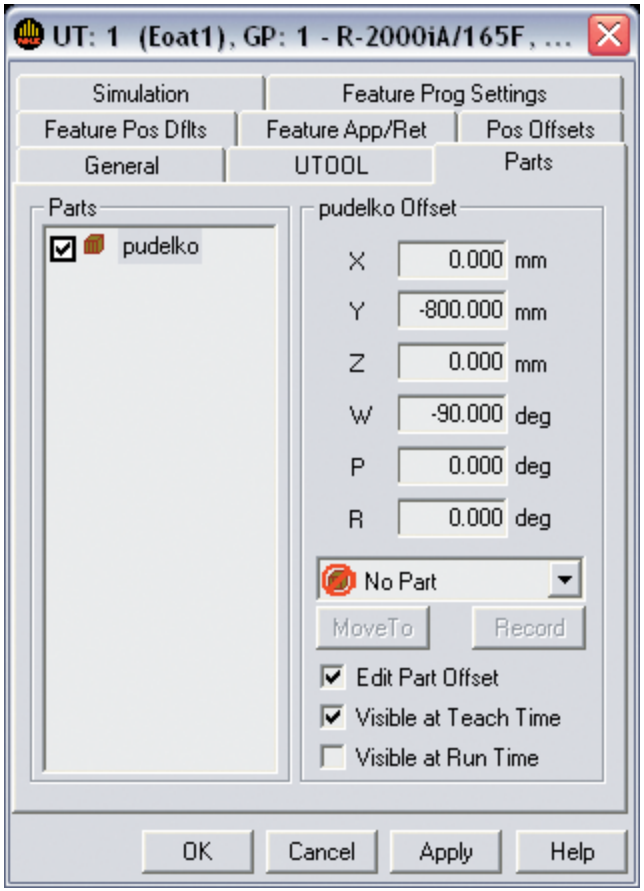
W ten sposób możemy rozbudowywać naszą wirtualną przestrzeń o dowolne elementy.



Każdy element, który znajduje się w przestrzeni projektu ma swoje właściwości.

Okno z właściwościami wywołujemy dwukrotnie klikając na dany element .

Konfiguracja chwytaka



Następnym celem jest konfiguracja chwytaka. W symulatorze są dostępne przykładowe chwytaki.

Na zakładce "UTOOL„ możemy ustawiać przesunięcia naszego robota.

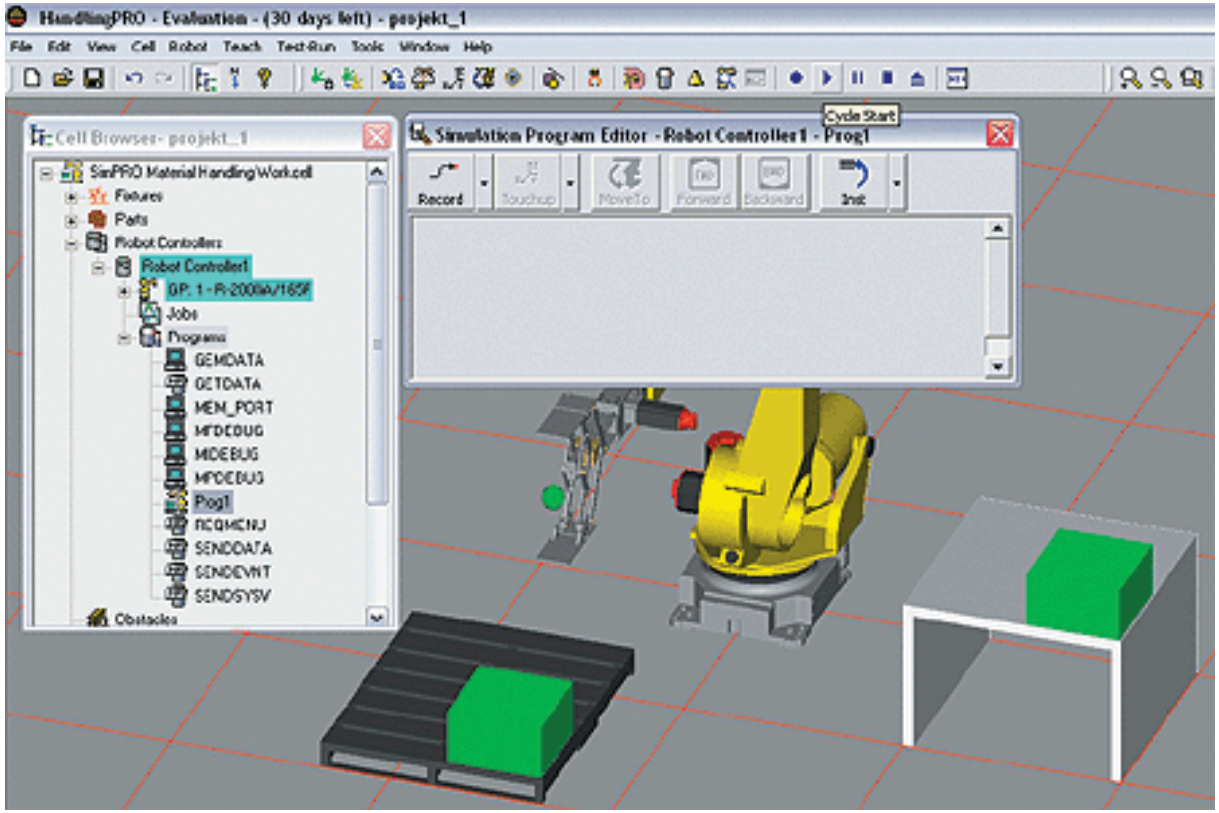
Na zakładce „Parts” można:

- stworzyć powiązania między chwytakami a elementami które będą przenoszone
- zdefiniować przesunięcie rezentacji w trakcie symulacji.

W zakładce "Simulation„ ustawiamy:

- pozycję chwytaka w pozycji zamkniętej
- pozycję chwytaka w pozycji otwartej

Wygląd przykładowej symulacji



Film1 – tutorial

Film2 – symulacja pracy

RoBOSS

http://sourceforge.net/projects/roboss/

RoBOSS

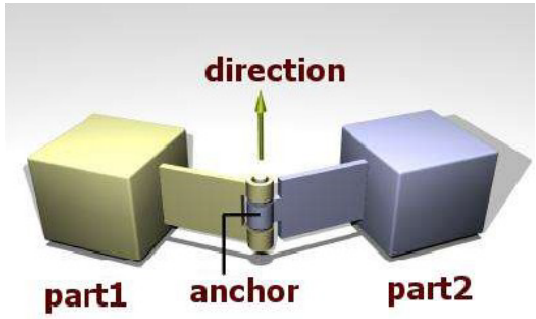
Narzędzie symulacyjne przeznaczone do wspomagania badań nad sterowaniem grupami robotów
- Roboty mobilne
- Przemysłowe linie produkcyjne
- Systemy transportowe

Przykłady i możliwości zastosowań
- Badanie metod współpracy robotów
- Badanie złożonych algorytmów sterowania grupami robotów
  - Metody sztucznej inteligencji
  - Systemy agentowe
- Badanie zachowań robotów w sytuacjach nieprzewidywalnych/kryzysowych
  - Symulacja awarii linii produkcyjnej
  - Symulacja awarii systemu transportowego/logistycznego wykorzystującego inteligentne roboty mobilne
  - Monitoring środowiska wykorzystującego roboty - badanie możliwości wykrywania sytuacji nietypowych i awaryjnych

RoBOSS

Definiowanie dowolnych typów i kształtów robotów

- części sztywnych z jakich robot jest zbudowany
- połączeń pomiędzy poszczególnymi częściami
- silników, w jakie wyposażony jest robot
- czujników/sensorów (kamera, kompas, gps, dalmierz ...)
- elementów wykonawczych (efektorów)
  (np. ramie wykonawcze, chwytak)



hinge

Realistyczna kinematyka i dynamika

Definiowanie dowolnego środowiska (własne modele świata i robotów)

Sterowanie programowe

- możliwość tworzenia programów sterujących w różnych językach programowania
- możliwość testowania programów sterujących

RoBOSS

Kontroler

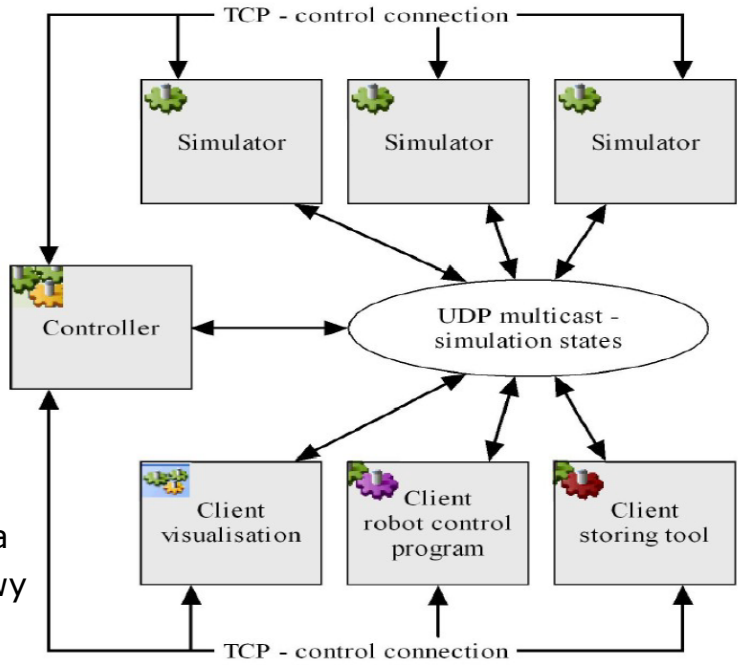
- Przechowuje definicję środowiska oraz robotów
- zarządza przebiegiem symulacji
- zarządza wymianą inf. przez symulatory

Symulatory

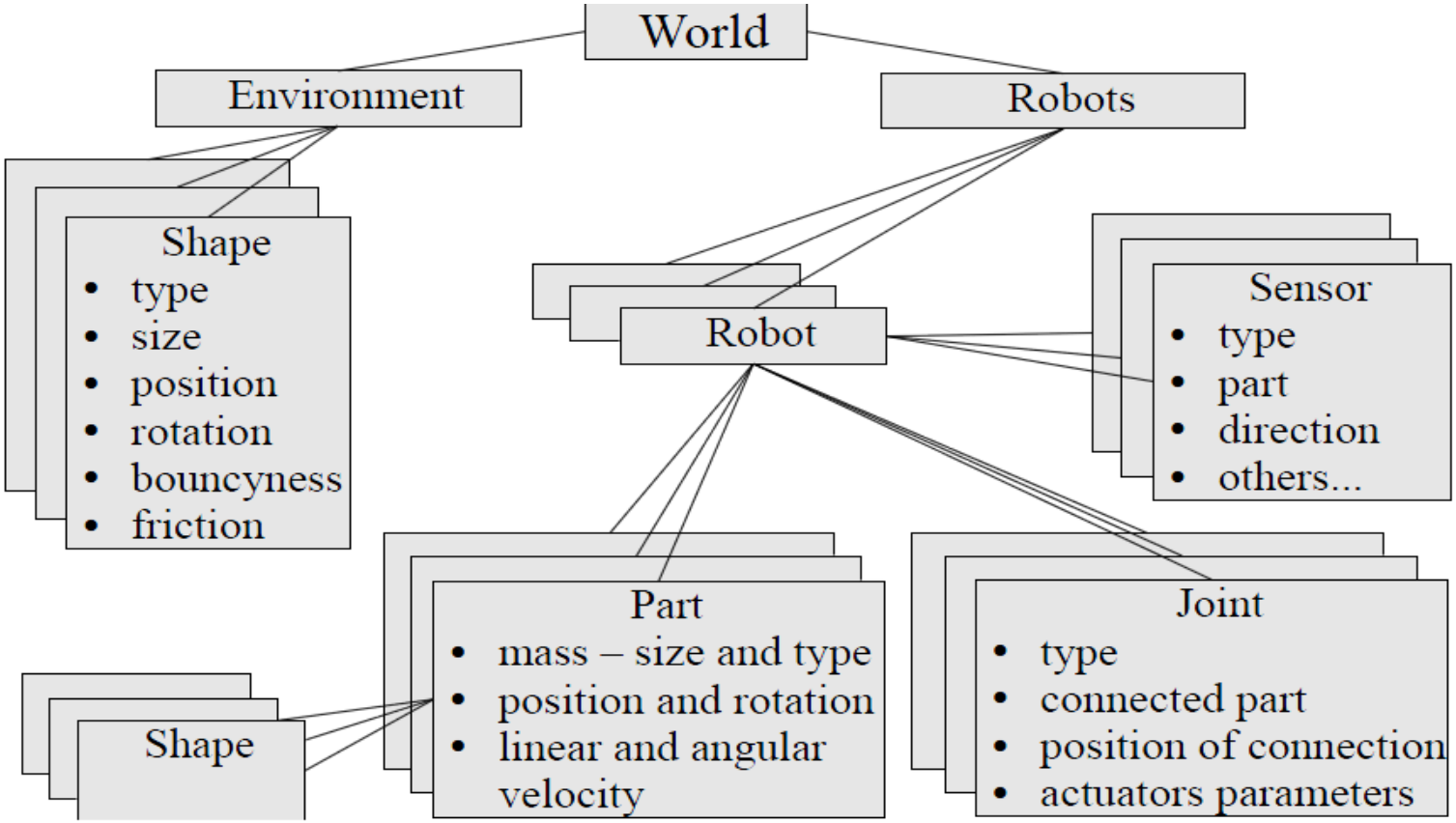
- Symulacja może odbywać się w środowisku rozproszonym
- Każdy symulator może realizować symulację odrębnego robota lub grupy robotów

Klienci

- Programy sterujące robotami
- mogą być pisane w dowolnym języku programowania
- komunikacja z symulatorem poprzez interfejs sieciowy
- Wizualizacja
- Programy monitorujące środowisko, zbierające dane o przebiegu symulacji



Schemat konstrukcji środowiska

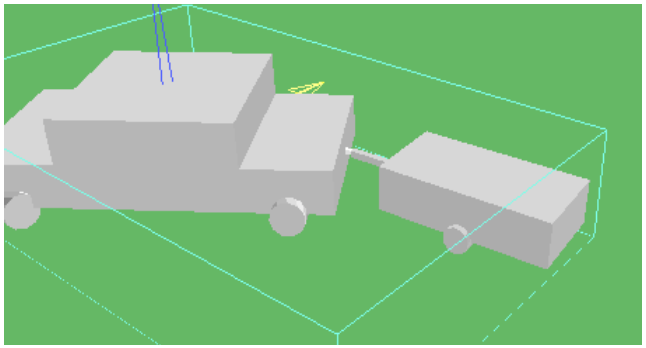


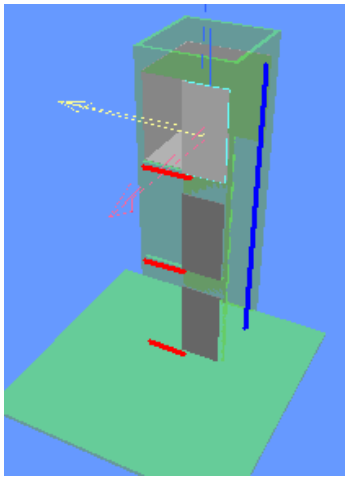
Język opisu modelu symulacyjnego

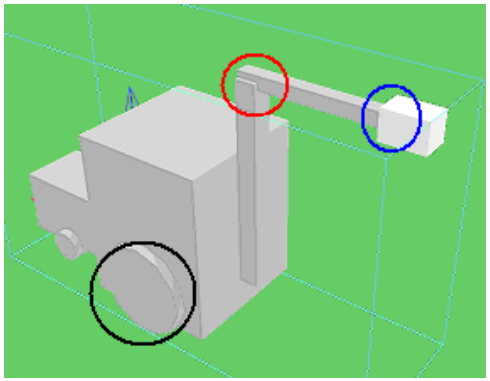
```
<?xml version="1.0" encoding="UTF-8"?>
<World name="nome" gravity=”9.81”>
<Robots>
<!-- definicje robotów -->
</Robots>
<Environment>
<!-- definicje elementów środowiska -->
</Environment>
</World>
```

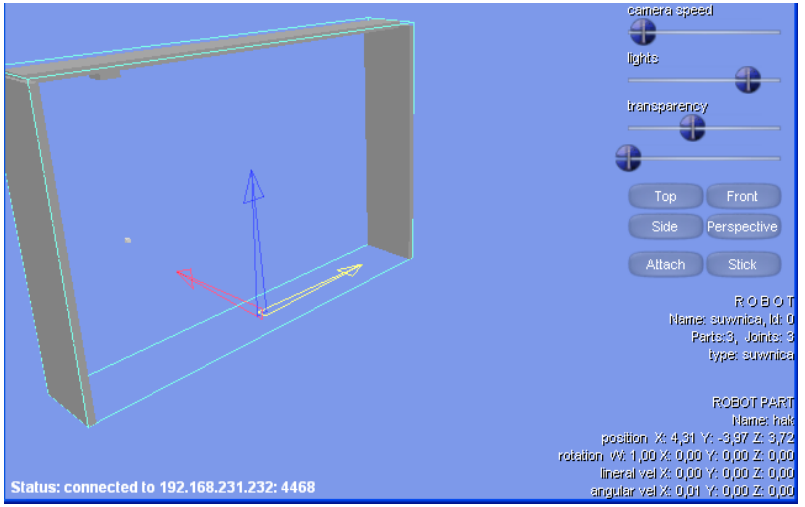
Połączenia w symulatorze ROBOSS

- Slider
- Lineal_Servo
- Angular_Servo
- Hinge
- Ball
- Driving_Wheel
- Universal
- Transmission









Parametry połączeń

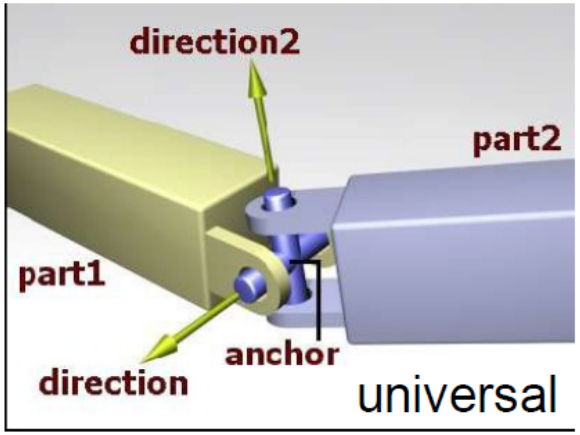
	Joint				Motor1				Motor2	
typ	anchor	direction	direction2	Stop	position	velocity	force	Stop2	velocity	force
ball	x									
slider		x		x		x	x			
lineral_servo		x		x	x	x	x			
hinge	x	x		x		x	x			
angular_servo	x	x		x	x	x	x			
universal	x	x	x	x		x	x	x	x	x
driving_wheel	x	x	x	x	x	x	x		x	x
transmission		x								

Parametry połączeń

ANCHOR
- anchor_x="0.5" anchor_y="-1" anchor_z="-0.25”
- Punkt zaczepienia połączenia
- Dla połączeń: *Ball, Hinge, Angular_Servo, Driving_Wheel, Universal*

DIRECTION
- direction_x="0" direction_y="1" direction_z="0"
- Oś połączenia
- Dla połączeń *Slider* i *Lineral_Servo* **-** kierunek przesunięcia
- Dla połączeń *Hinge, Angular_Servo, Universal, Driving_Wheel, Transmission* - oś obrotu

DIRECTION2
- direction2_x="0" direction2_y="1" direction2_z="0
- Oś obrotu drugiego silnika
- Wektor prostopadły do DIRECTION
- Dla połączeń: *Universal, Driving_Wheel*



STOP
- <Stop hi="5.5" lo="-5.5" />
- Parametr opcjonalny, Stosowany dla potrzeb konkretnego modelu
- Dla połączeń *Slider i Lineral_Servo* ma charakter liniowy
- Dla połączeń *Hinge, Angular_Servo, Universal, Driving_Wheel* ma wartość kąta w radianach

Parametry połączeń silników

MOTOR1 - POSITON

- communicator.robots[0].joints[0].motorDesiredPosition = 5 a
- Parametr sterujący
- Powinien się zawierać pomiędzy **Stop lo**, a **Stop hi**
- Dla połączenia **Lineral_Servo** parametr liniowy
- Dla połączeń **Angular_Servo** i **Driving_Wheel** parametr kątowy wyrażony w radianach

MOTOR1 - VELOCITY

- Motor.const_velocity="5"
- Dla połączeń *Lineral_Servo, Angular_Servo i Angular_Servo*
- Prędkość maksymalna silnika
- communicator.robots[0].joints[4].motorDesiredVelocity = 5
- Parametr sterujący – prędkość docelowa
- Dla połączeń *Slider, Hinge i Universa*l

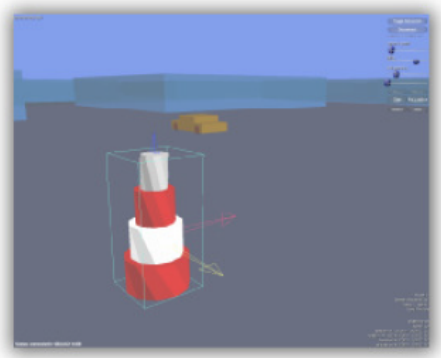
```
<Joint name="unikatowanazwa" type=""
part1="" part2=""
direction_x="" direction_y="" direction_z=""
direction2_x="" direction2_y="" direction2_z=""
anchor_x="" anchor_y="" anchor_z="">
   <Stop hi="" lo="" />
   <Motor const_position="" const_velocity=""
const_force=""/>
   <Stop2 hi="" lo="" />
   <Motor2 const_velocity="" const_force=""/>
</Joint>
```

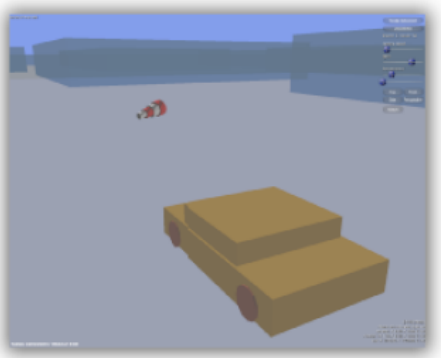
MOTOR – FORCE

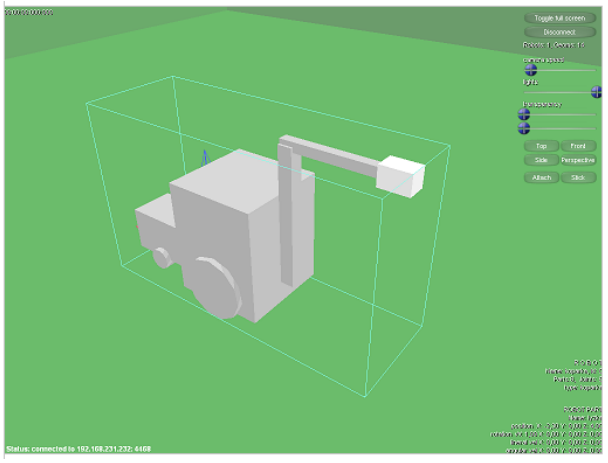
- <Motor const_force="20"/>
- Moc silnika
- Dla połączeń *Hinge, Angular_Servo, Universal, Driving_Wheel, Slider i Lineral_Servo*

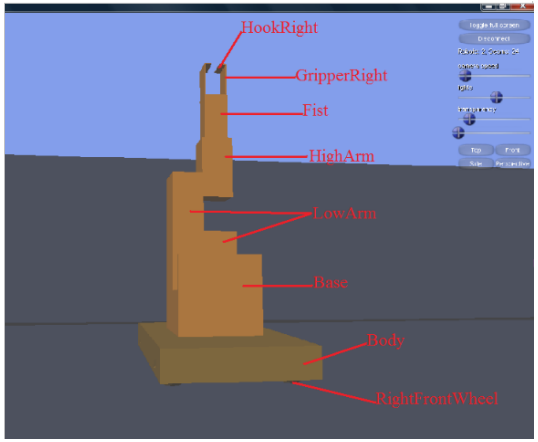
RoBOSS - zastosowania

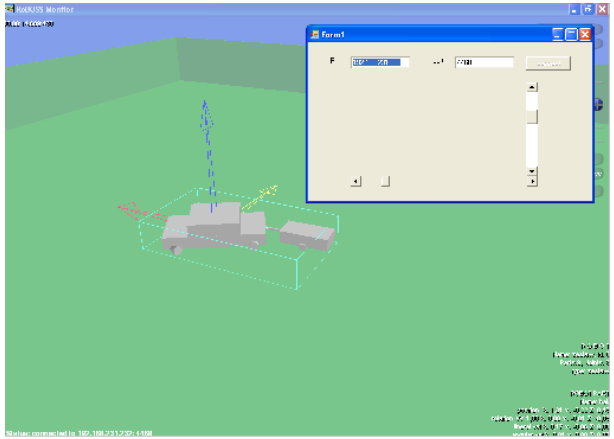
- możliwość przyłączania elementów tworzących dynamiczne konstrukcje ruchome
- połączenie typu ball - łączenie samochodu z przyczepą
- Możliwość konstrukcji zaawansowanych wielobryłowych elementów robota











COSIMIR

http://www.axicont.com/Cosimir.htm
http://www.mpl.pl/

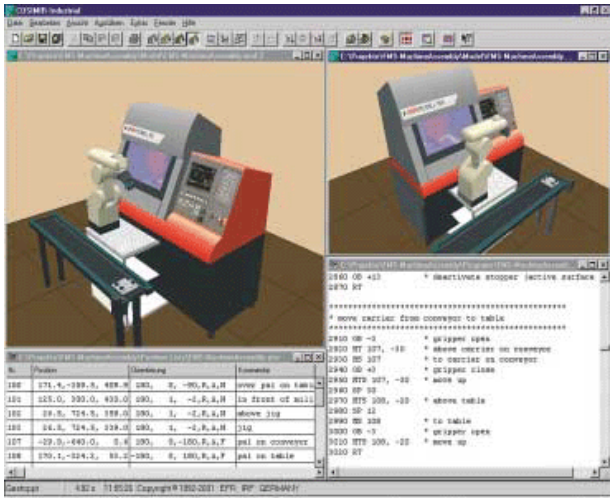
Cosimir

Symulator firmy Mitsubishi Electric

Wykorzystywany w przemyśle do projektowania i symulacji instalacji zrobotyzowanych

Umożliwia:
- tworzenie i wykonywanie testów na wirtualnej taśmie produkcyjnej (aktualnie istniejącej, lub nowej-projektowanej)
- tworzenie i testy oprogramowania sterującego robotem

Bogatej asortymentem produktów (biblioteka)
- roboty,
- przenośniki taśmowe,
- chwytaki



Cosimir -roboty

Wykorzystywany głównie dla robotów przemysłowych Mitsubishi Electric:

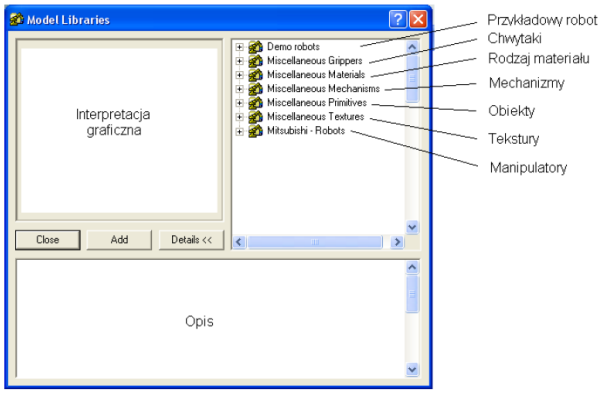
- RV-1A
- RV-2AJ

Programowanie robotów w językach
- MELFA BASIC IV
- MOVEMASTER COMMAND.

Oprogramowanie napisane na potrzeby symulacji można przenieść do rzeczywistych robotów.

Przykładowy projekt z wykorzystaniem symulatora

Pierwszym krokiem pracy z symulatorem jest ustawienie powierzchni podłogi i jej parametrów (wymiar, materiał, położenie)

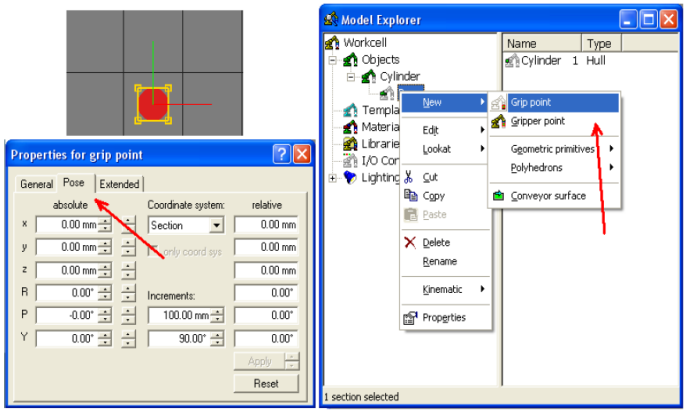


Do projektu dodajemy elementy biorące udział w symulacji (z bibliotek programu)

- robot,
- przesuwnik taśmowy,
- manipulowany obiekt,
- elementy otoczenia.

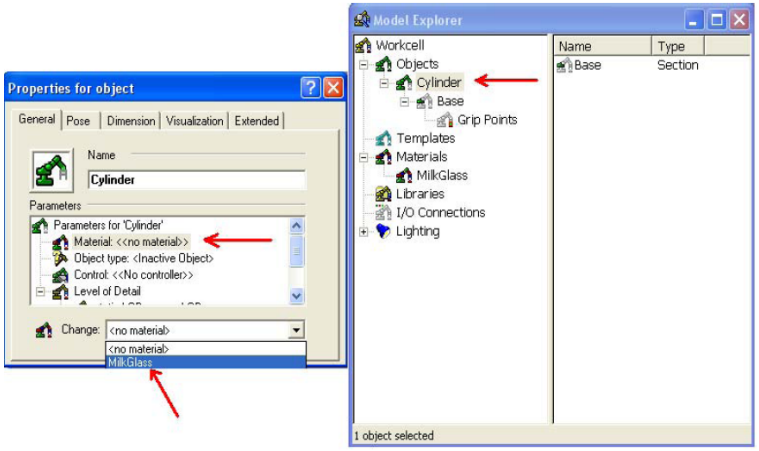
Ustawianie punktu uchwytu

- służy do powiązania między sobą różnych elementów.
- określa punkty w których będą mogły być realizowane funkcje połączeń
- realizuje :
  - złapanie przedmiotu przez chwytak,
  - poruszanie się elementu po przenośniku



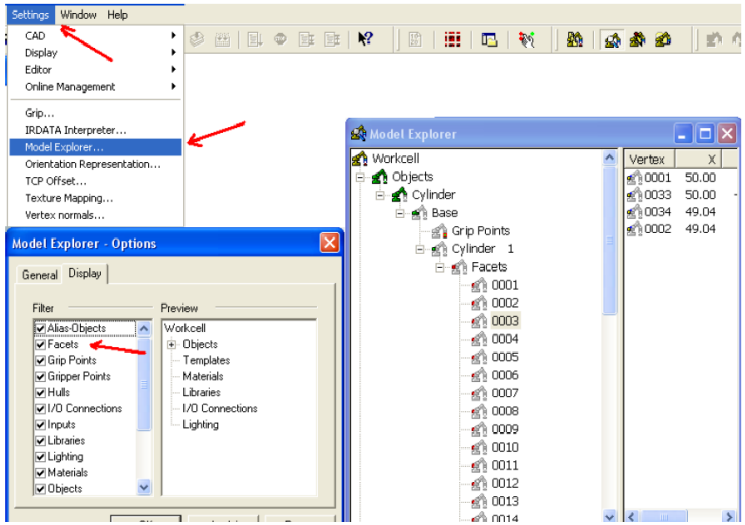
Projekt z wykorzystaniem symulatora: c.d.

Przypisanie materiału do obiektów: uwzględnienie szczegółów występujących w rzeczywistości. (możliwość dodania swoich materiałów.)

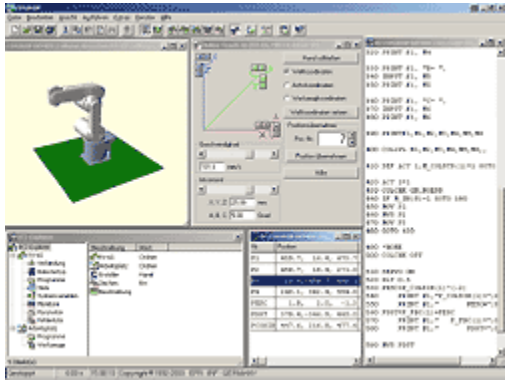


Zmiana ilości wyświetlanych elementów.

Nieistotna z punktu widzenia procesów sterowania manipulatorem. Poprawiająca percepcję symulacji.



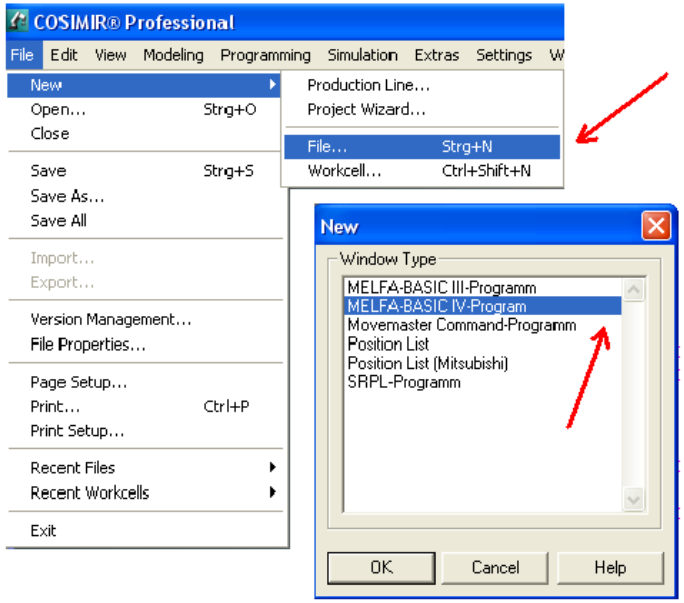
Przykładowy projekt – c.d.



Połączenie robota z urządzeniami peryferyjnymi:
- Ustalenie zależności pomiędzy poszczególnymi elementami.
- Definicje sygnałów wejściowych i wyjściowych.

Tworzenie kodu ruchu robota w języku „MELFA-BASIC IV”
Wymaga podpisywania linii kodu.
Prawidłowe komendy wyświetlane są kolorem niebieskim, wartości liczbowe – różowym.

Podstawowe komendy:
- *mov(x,y,z)* - ruchu do punktu o współrzędnych x.y.z.
- *M_out(nr wyjścia)=stan* – ustawienie wyjścia na stan 1 lub 0.
- *wait M_in(nr wejścia)=stan* - oczekiwanie na zmianę stanu wejścia (1 lub 0).
- *MOV*- robot porusza się do wyznaczonej pozycji z łączną interpolacją.
- *MVS* - robot porusza się do wyznaczonej pozycji interpolacją liniową
- *MVR* - ruch z interpolacją kołową.
- *GOTO linia* - komenda skoku do linii



Uruchomienie symulacji:
- ostatni etap tworzenia aplikacji.
- łączenie w całość wszystkich elementów projektu.
  - Interpretacji graficznej
  - programu

symulacja

MICROSOFT ROBOTICS DEVELOPER STUDIO

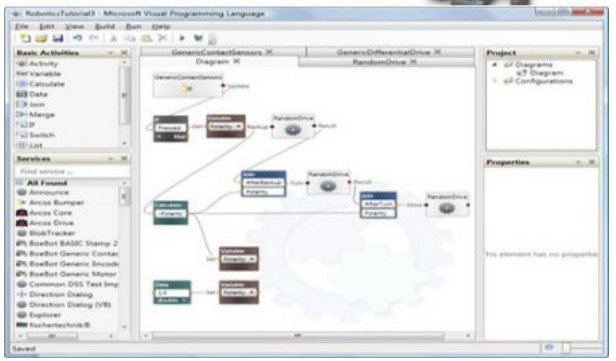
http://www.microsoft.com/robotics/

Microsoft Robotics Developer Studio

Platforma programistyczna przeznaczona dla systemów Windows, umożliwiająca tworzenie oprogramowania dla robotów .

Środowisko kompatybilne z następującymi rozwiązaniami sprzętowymi:

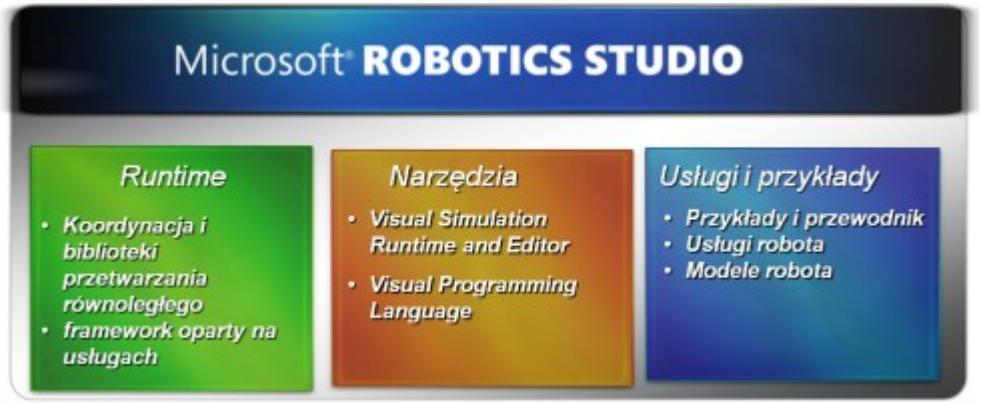
- Fischertechnik,
- Roomba
- Lego Mindstorms NXT.



Najważniejsze cechy środowiska:

- tworzenie aplikacji zorientowanych na usługi, współpracujących z szeroką gamą sprzętu używanego w robotyce.
- graficzny język programowania (Visual Programming Language)
- Możliwość symulowania działania aplikacji korzystając z realistycznych modeli w wirtualnym świecie symulacji.
- Mechanizm Concurrency and Coordination Runtime (CCR) ułatwiający asynchroniczną wymianę danych.
- Możliwość budowania aplikacji niezależnych sprzętowo. Testy na procesorach 8, 16 i 32 bitowych oraz wielordzeniowych.
- Otwartość platformy. Możliwość dodawania nowych bibliotek oraz serwisów przez zewnętrznych producentów sprzętu.
- Komunikacja pomiędzy robotem a PC poprzez port szeregowy, Bluetooth, 802.11, lub RF. Aplikacja może być również uruchamiana na komputerze pokładowym robota.

Trzy podstawowe elementy MS RS



1. Runtime - silnik całego rozwiązania, który pozwala na oprogramowanie robota
2. Narzędzia - pozwalają oprogramować urządzenie
3. Usługi i przykłady - dokumentacja wraz z przykładami dotycząca dostarczonych usług

Runtime składa się z dwóch najważniejszych elementów:

- Concurrency and Coordination Runtime – upraszcza pisanie asynchronicznych aplikacji dzięki możliwości uniknięcia ręcznego sterowania wątkami, blokadami, semaforami, itd.
- Decentralized Software Services – model aplikacji opartych na usługach (SOA - services oriented application), który wspiera model programowania REST (Representational State Transfer)

MSDS – c.d.

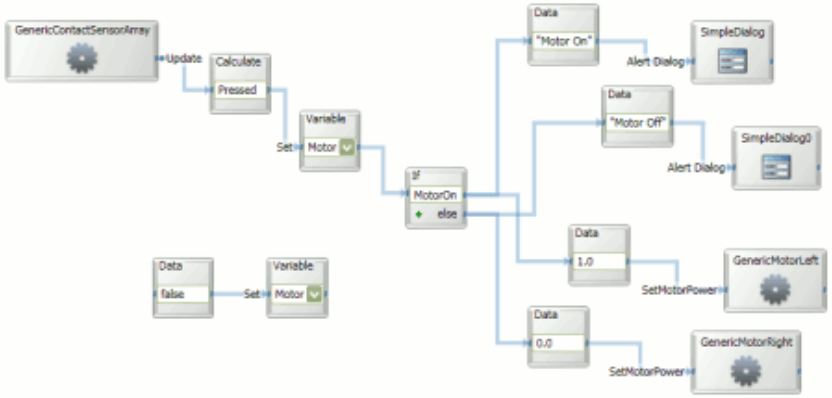
Usługi zdefiniowane na pewnym poziomie abstrakcji mogą reprezentować:

- Sprzęt – sensory, aparaty, etc.
- Oprogramowanie – interfejs użytkownika, miejsce składowania danych, etc.
- Agregacje – mash-upy, połączenie sensorów, etc.

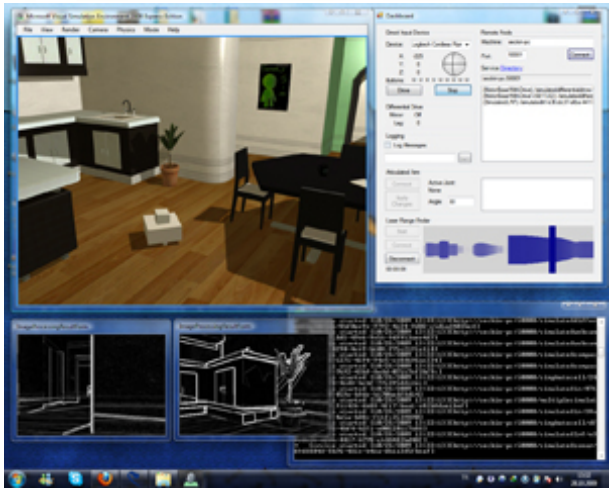
Separacja stanu usługi od jej zachowania - możliwość ponownego użycia poprzez kompozycję.

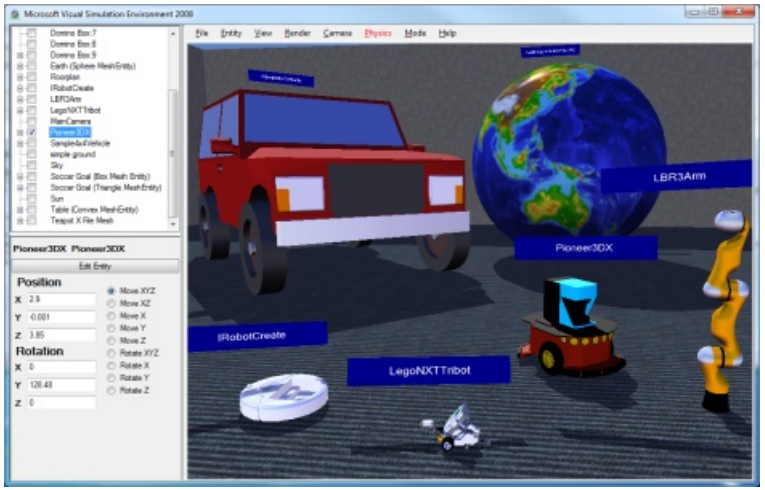
Aplikacja dla robota jest zestawem kilku elementów.

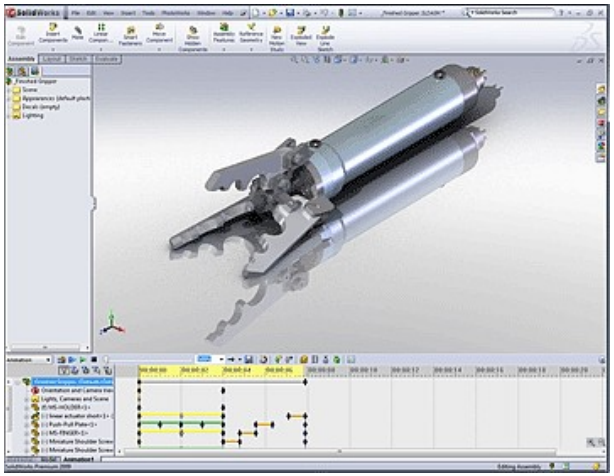
- urządzenia, sensory
- model zachowania urządzeń (orkiestracja)
- prezentacja dla użytkownika

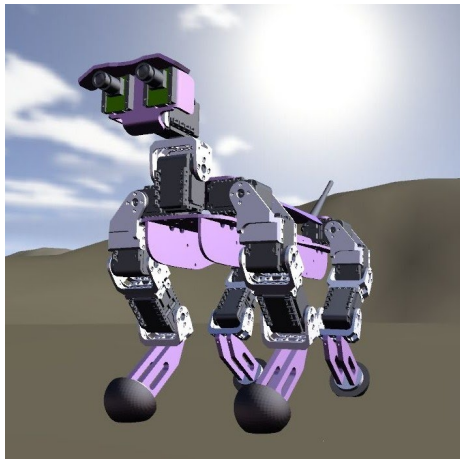


MSDS – przykłady









Bibliografia:

- S.Jeżewski, M.Łaski - Przegląd i porównanie środowisk symulacyji robotów mobilnych – Automatyka 2009, t13, z3
- Cosimir – Instrukcja obsługi programu
- K.Szopa - Symulatory Robotów mobilnych, prezentacja - IMiIP, SIP, 2010.
- http://tech.wp.pl/kat,1,page,2,title,Budujemy-robota-czesc-I-wstep-do-Robotics-Studio,wid,10208658,wiadomosc.html
- Cosimir - Getting Started – Short introduction into 3D Simulation and Offline Programming of robot-based workcells with COSIMIR – EFR – IRF – Nov 1 2000
- Front, Folwarczny - MRDS- Prezentacja – IMiIP, SysRA
- A.Widomski - Przykłady wykorzystania połączeń w RoBOSS, IMiIP, SysRA
- E.Nawrecki – Planowanie agentowe w zmieniającym się dynamicznie środowisku z uwzględnieniem sytuacji kryzysowych.- Grant KBN: 3 T11C 025 27
- P. Przydatek, M.Kaczmarek - Programowanie robotów przemysłowych Fanuc Robotics – Astor sp. Z o.o. - http://www.astor.com.pl/biuletyn-automatyki/archiwum/rocznik-2006/nr-48-32006-/473.html