

IDŹ DO

PRZYKŁADOWY ROZDZIAŁ



SPIS TREŚCI

KATALOG KSIĄŻEK

KATALOG ONLINE

ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK

DODAJ DO KOSZYKA

CENNIK I INFORMACJE

ZAMÓW INFORMACJE
O NOWOŚCIACH

ZAMÓW CENNIK

CZYTELNIA

FRAGMENTY KSIĄŻEK ONLINE

CATIA v5. Modelowanie i analiza układów kinematycznych

Autor: Marek Wyleżół

ISBN: 83-246-0678-5

Format: B5, stron: 384



CATIA v5 to wielomodułowy system CAD/CAM wspomagający projektowanie mechaniczne – od szkiców, poprzez rysunki detali, zdefiniowanie zespołów i wygenerowanie dokumentacji wykonawczej. We współczesnym procesie projektowania stworzenie prototypu urządzenia poprzedzane jest wykonaniem jego wirtualnych odpowiedników, nazywanych makietami. Na takich makietach przeprowadza się symulacje i obliczenia mające na celu zoptymalizowanie konstrukcji urządzenia. Proces taki nazywany jest wirtualnym prototypowaniem.

Czytając książkę "CATIA v5. Modelowanie i analiza układów kinematycznych" zdobędziesz wiedzę na temat jednego z etapów wirtualnego prototypowania – przeprowadzania symulacji kinematycznych. Poznasz DMU Kinematics Simulator – moduł systemu CATIA v5 wykorzystywany właśnie do tego celu. Dowiesz się, w jaki sposób osadzać w jego przestrzeni roboczej modele elementów składowych i tworzyć z nich modele urządzeń. Nauczysz się prowadzić symulacje za pomocą poleceń systemu oraz formuł i reguł. Poznasz także sposoby analizy trajektorii ruchu, zajmowanej przestrzeni oraz kolizji.

- Interfejs użytkownika modułu DMU Kinematics Simulator
- Operacje na plikach
- Import modeli do modułu
- Modelowanie mechanizmów
- Symulacje z użyciem poleceń
- Rejestracja symulacji
- Zastosowanie formuł i reguł do symulacji ruchu
- Analiza kinematyczna

Poznaj metody pracy stosowane we współczesnych biurach projektowych

Wydawnictwo Helion
ul. Kościuszki 1c
44-100 Gliwice
tel. 032 230 98 63
e-mail: helion@helion.pl



Spis treści

Wstęp	5
Rozdział 1. Wprowadzenie do modułu DMU Kinematics Simulator	11
Interfejs modułu DMU Kinematics Simulator	11
Przegląd narzędzi programowych	12
Operacje dyskowe	17
Zapisywanie plików	18
Otwieranie plików	20
Rozdział 2. Rozpoczęcie pracy z modułem DMU Kinematics	21
Osadzanie modeli elementów składowych mechanizmów w przestrzeni roboczej	23
Import złożenia wykonanego w środowisku modułu Assembly Design	23
Import pojedynczych modeli do środowiska modułu DMU Kinematics	36
Modelowanie elementów składowych mechanizmu w środowisku modułu DMU Kinematics	50
Rozdział 3. Przykłady modelowania mechanizmów	63
Mechanizm kątowej przekładni ciernej	65
Mechanizm chwytaka	83
Mechanizm korbowy	101
Mechanizm krzywkowy z popychaczem rolkowym	113
Mechanizm dwukrzywkowy	126
Mechanizm krzywki śrubowej	135
Mechanizm wahacza wleczonego	154
Mechanizm przekładni zębatej walcowej	170
Mechanizm zębatkowy	180
Mechanizm przekazania napędu	192
Rozdział 4. Symulacje z użyciem komend	205
Symulacje ruchu z użyciem komend	206
Symulacje bezpośrednie	210
Symulacje „na życzenie”	215
Rejestracja symulacji	224
Przykład rejestrowania symulacji	228
Tworzenie sekwencji	234
Nagrywanie powtórek	249
Nagrywanie i zapisywanie symulacji do postaci formatów plików zewnętrznych	255

Rozdział 5. Symulacje z zastosowaniem formuł i reguł	261
Zastosowanie formuł do symulacji ruchu	262
Zastosowanie formuł do symulacji ruchu posuwowego	263
Zastosowanie formuł do symulacji ruchu obrotowego	279
Zastosowanie formuł do symulacji ruchu złożonego	285
Zastosowanie formuł i funkcji matematycznych do symulacji ruchu złożonego	291
Zastosowanie reguł do symulacji ruchu	294
Zastosowanie reguł do symulacji ruchu złożonego	297
Zastosowanie wykresów prędkości do symulacji ruchu	305
Rozdział 6. Analizy mechanizmów	317
Analiza trajektorii	317
Analiza zajmowanej przestrzeni	326
Analiza odległości	334
Analiza odległości podczas symulacji ruchu z użyciem komend	339
Analiza odległości podczas edycji symulacji	342
Analiza kolizji	348
Zastosowanie analizy kolizji podczas symulacji ruchu z użyciem komend	354
Zastosowanie analizy kolizji podczas edycji symulacji	360
Analiza prędkości i przyspieszeń	363
Bibliografia	373
Skorowidz	375

Rozdział 5.

Symulacje z zastosowaniem formuł i reguł

W poprzednim rozdziale zajmowaliśmy się realizacją symulacji ruchu mechanizmów z użyciem komend. Tego rodzaju rozwiązanie pozwala na „ręczne” sterowanie sposobem realizacji ruchu. Dzięki temu uzyskujemy wiele informacji związanych np. z postacią ruchu, z ustaleniem możliwych zakresów przesuwu czy obrotu członów tworzących dany łańcuch kinematyczny itp.

Natomiast, stosując wspomniany sposób realizacji symulacji ruchu, nie mamy żadnego wpływu na ważny parametr ruchu, jakim jest **prędkość**. Używając np. suwaków czy pokręteł, nie mamy kontroli nad prędkością ruchu, w przypadku gdyby ważna była analiza ruchu jednostajnego (o stałej prędkości liniowej czy kątowej).

Namiastką takiej możliwości jest stosowanie symulacji „na życzenie”, gdzie można dokładnie zaplanować postać ruchu i wykonać płynną jego symulację. Ale również w tym przypadku jesteśmy zdani na ustalenie przez system domyślnej prędkości ruchu (owszem, możemy planować całkowity czas realizacji ruchu, ale bez podziału na poszczególne etapy).

Więcej możliwości daje nam tu stosowanie sekwencji, gdzie można dość dokładnie zaplanować czas trwania danego fragmentu ruchu. Wymaga to jednak wygenerowania odpowiednio dużej ilości symulacji dla każdego fragmentu ruchu.

Rozwiązaniem problemów z dokładnym sterowaniem parametrami ruchu, a w szczególności jego prędkością, jest stosowanie **formuł matematycznych i reguł logicznych**. Możliwość taką zawdzięczamy budowie systemu *CATIA*, który jest wielomodułowym systemem zintegrowanym. A więc można praktycznie bez ograniczeń stosować narzędzia charakterystyczne dla różnych modułów w jednym projekcie np. mechanizmu.

Tak więc, stosując moduł *DMU Kinematics Simulator*, możemy jednocześnie stosować narzędzia programowe charakterystyczne np. dla modułu *Knowledge Advisor*. A dokładnie chodzi o możliwość stosowania reguł do opisu ruchu. To właśnie z ich pomocą uzyskujemy możliwość uwzględnienia podziału ruchu na odcinki czasowe oraz ustalenia prędkości realizacji ruchu wraz z uwzględnieniem jego rodzaju.



Pełne zrozumienie niniejszego rozdziału wymaga podstawowej znajomości modułu *Knowledge Advisor*.

W przypadkach prostszych, gdy mamy do czynienia z realizacją ruchu, gdzie nie wyróżniamy poszczególnych jego etapów czasowych, oraz gdy w ramach danego etapu czasowego nie musimy sterować prędkością ruchu wielu członów, wystarczy nam zastosowanie formuł matematycznych.

W ramach stosowania formuł i reguł możemy dodatkowo korzystać z możliwości, jakie daje nam stosowanie parametrów oraz funkcji obliczeniowych systemu.

Zupełnie inną klasą zagadnień, które zostały omówione w końcowej części niniejszego rozdziału, jest stosowanie tzw. **wykresów prędkości**, gdzie za pomocą elementów graficznych (np. proste, łuki, krzywe splajnowe) można planować sposób poruszania się mechanizmu, a następnie z użyciem stosownych funkcji dokonać zamiany postaci graficznej na matematyczną.

Podstawowym parametrem systemu, jaki musimy stosować podczas realizacji symulacji ruchu z użyciem formuł czy reguł, jest parametr o nazwie **KINTime**. Jego jednostką jest czas. Przy standardowych ustawieniach systemu jest to sekunda (s). W praktyce w parametrze tym zawarty jest czas realizacji danej symulacji. Jego szczegółowe zastosowanie zostało pokazane w ramach opisów poszczególnych przykładów ćwiczeniowych.

Do realizacji symulacji ruchu z użyciem formuł czy reguł moduł *DMU Kinematics Simulator* został wyposażony w narzędzie programowe *Simulation with Laws* (patrz rozdział 1.).

Szczegółowe zagadnienia związane z realizacją symulacji ruchu z użyciem formuł i reguł, a także z użyciem parametrów (systemowych oraz użytkownika), jak i funkcji specjalnych systemu, zostały przedstawione w poszczególnych punktach niniejszego rozdziału.



Zdefiniowanie formuł i (lub) reguł jest również warunkiem koniecznym efektywnej realizacji wielu analiz kinematycznych.

Zastosowanie formuł do symulacji ruchu

W punkcie tym zawarto praktyczne informacje związane z zastosowaniem formuł matematycznych do realizacji symulacji prostego ruchu posuwowego, obrotowego i ruchu złożonego. Poza tym przedstawiono przykład realizacji ruchu z użyciem funkcji matematycznych, jakie oferuje nam system *CATIA*.

Zastosowanie formuł do symulacji ruchu posuwowego

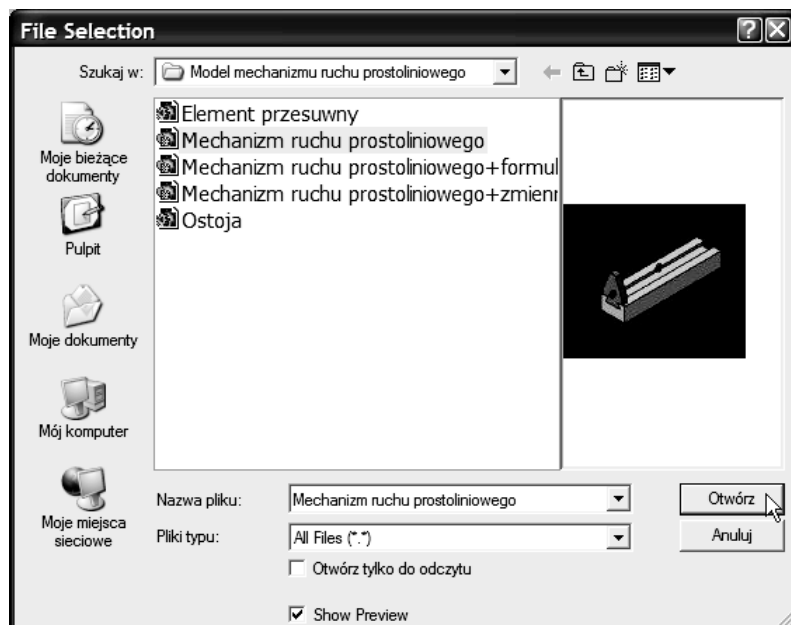
Zastosowanie formuł matematycznych do realizacji symulacji ruchu posuwowego poznamy z użyciem przygotowanego przykładu prostego mechanizmu.

W tym celu, otwierając katalog *Modele/Rozdział 5/Model mechanizmu ruchu prostoliniowego* (z załączonej płyty CD-ROM), wskazujemy plik *Mechanizm ruchu prostoliniowego.CATProduct*.

W oknie podglądu okna *File Selection* widzimy postać modelu mechanizmu (rysunek 5.1).

Rysunek 5.1.

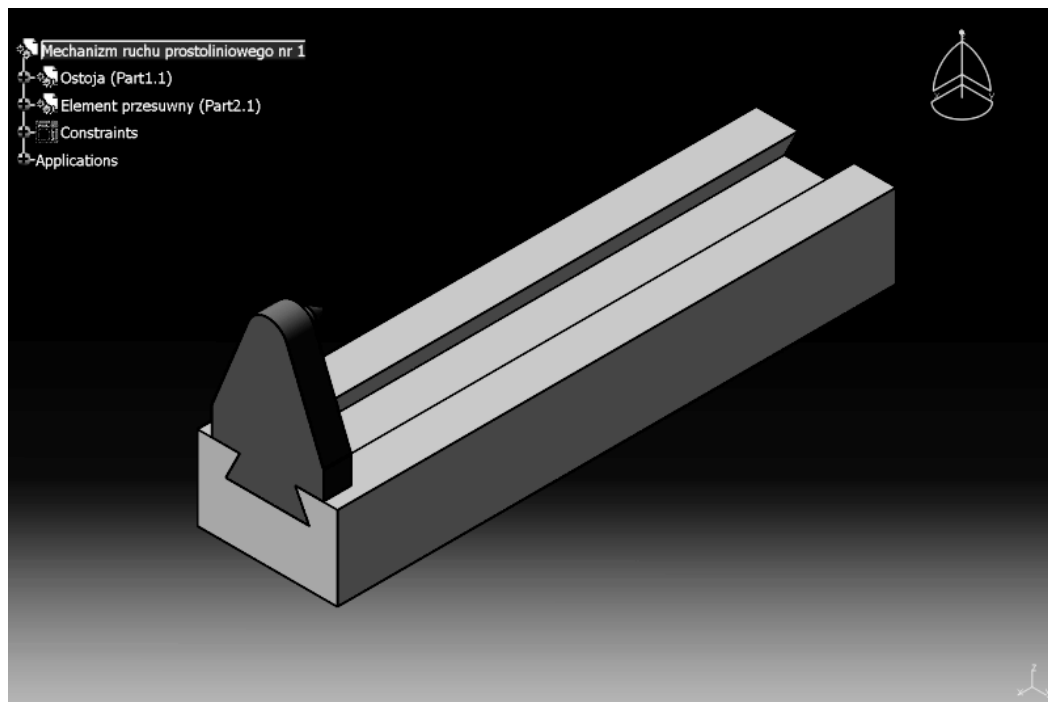
Widok zawartości okna *File Selection* podczas otwierania pliku *Mechanizm ruchu prostoliniowego.CATProduct*



Działanie mechanizmu jest bardzo proste. *Element przesuwny* wykonuje ruch posuwisty w prowadnicy *Ostoi* (rysunek 5.2). Model mechanizmu posiada jeden więz sterowany, typu *Prismatic*. Sterowaniu podlega długość ruchu (odległość, droga, jaką przebywa model).

Zakładamy, że chcemy, aby realizowany ruch był ruchem jednostajnym prostoliniowym. A więc prędkość ruchu powinna być stała i określona przez użytkownika. Czas ruchu również ma zostać określony przez użytkownika.

Zanim zajmiemy się utworzeniem stosownej formuły matematycznej, określającej prędkość ruchu, należy dokładnie zrozumieć dalsze czynności. Dla wytłumaczenia sposobu definiowania formuły posłużymy się wzorem na prędkość ruchu jednostajnego prostoliniowego: **prędkość liniowa = droga/czas**.



Rysunek 5.2. Widok postaci mechanizmu ruchu jednostajnego prostoliniowego

Jak już wspomnieliśmy, model mechanizmu zawiera jeden sterowalny więz, gdzie zmianie podlega długość ruchu modelu. A więc przekształćmy wzór na obliczanie prędkości, tak aby obliczyć z niego drogę. Otrzymamy wówczas zależność: **droga = czas × prędkość liniowa**.

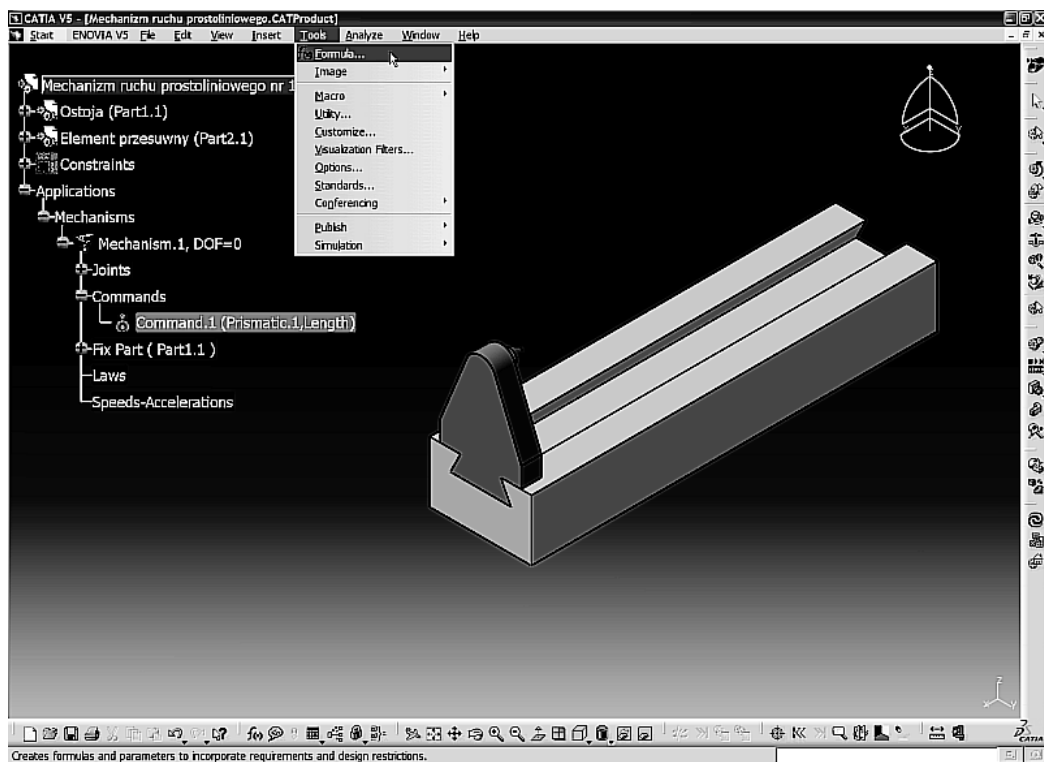
A więc, aby sterować długością drogi, jaką przebywa model, należy dysponować wartościami **czasu i prędkości**. W przytoczonym wzorze brakuje nam więc wartości dwóch wielkości: **czasu i prędkości**. Jak już wspomniano, czas realizacji ruchu zostanie zawarty w parametrze *KINTime* (wartość będzie zależała od użytkownika), natomiast prędkość ruchu musi zostać zdefiniowana za pomocą odpowiedniej wartości i jednostki (*mm/s*) przez użytkownika.

Do tego celu zastosujemy stosowną formułę matematyczną.

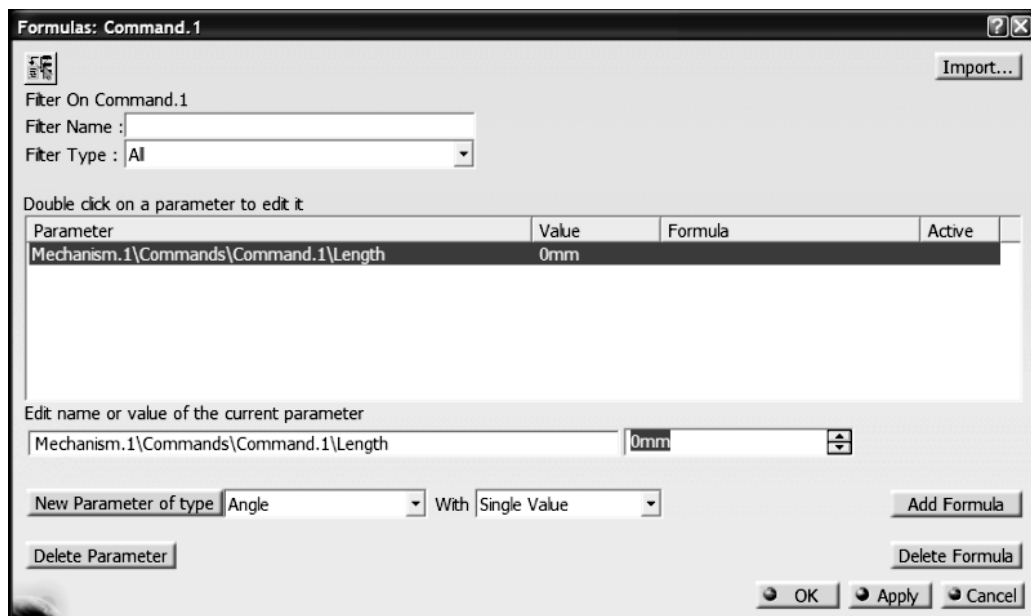


Utworzenie formuły dotyczącej realizacji ruchu mechanizmu jest dla systemu sygnałem uaktywniającym polecenie *Simulation with Laws*.

Aby formuła, jaką chcemy utworzyć, dotyczyła obliczania drogi, jaką ma przebyć w danym czasie i z daną prędkością model, przed uaktywnieniem polecenia *Formuła* klikamy myszą komendę *Command.1 (Prismatic.1, Length)* w drzewie topologicznym. Następnie za pomocą polecenia w menu *Tools\Formuła* (lub za pomocą stosownej ikony w belce narzędziowej, domyślnie umieszczonej w dolnej części ekranu, rysunek 5.3) uruchamiamy okno dialogowe *Formuła* (rysunek 5.4).



Rysunek 5.3. Uruchomienie polecenia *Formula*

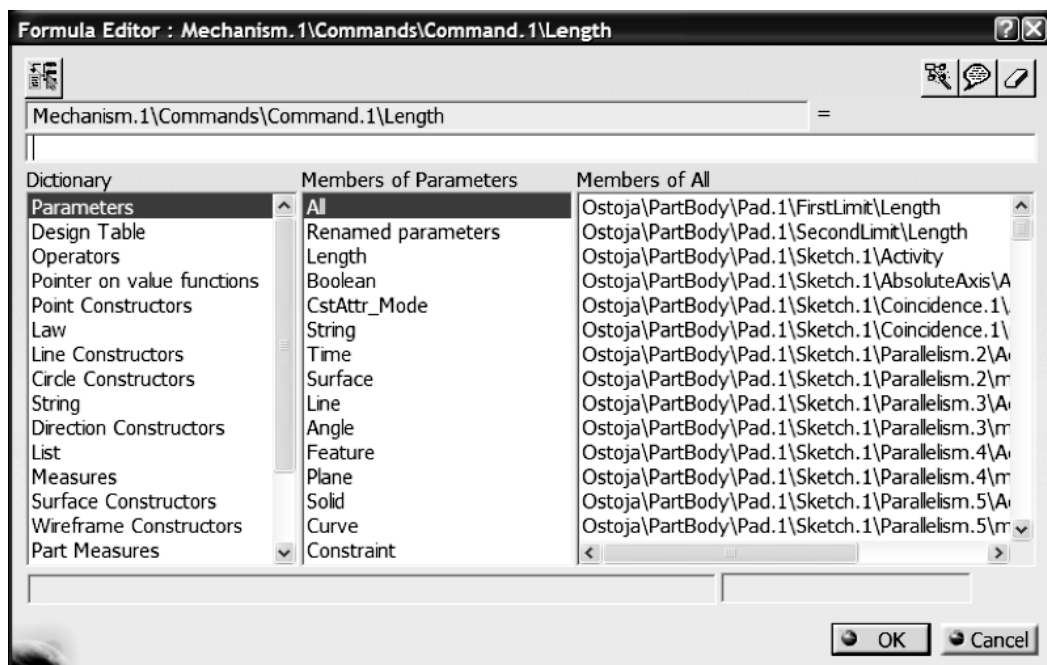


Rysunek 5.4. Widok zawartości okna dialogowego *Formulas*

W polu *Parameter* okna *Formulas* widzimy więc nazwę sterowania (komendy), której wartość chcemy uzależnić od czasu i prędkości. W polu *Value* widzimy, że domyślną jednostką odległości jest *mm*. Widoczna nazwa sterowania (*Mechanism.1\Commands\Command.1\Length*), które jednocześnie oznacza drogę, jest lewą stroną formuły (równania), które utworzymy.

W celu utworzenia formuły klikamy myszą przycisk *Add Formula* (rysunek 5.4).

Efektem naszego działania jest uaktywnienie się okna dialogowego *Formula Editor* (rysunek 5.5). Widzimy następujący układ formuły: *Mechanism.1\Commands\Command.1\Length = ...*. Naszym zadaniem jest wypełnienie prawej strony równania (zgodnie z równaniem: **droga = czas × prędkość liniowa**).

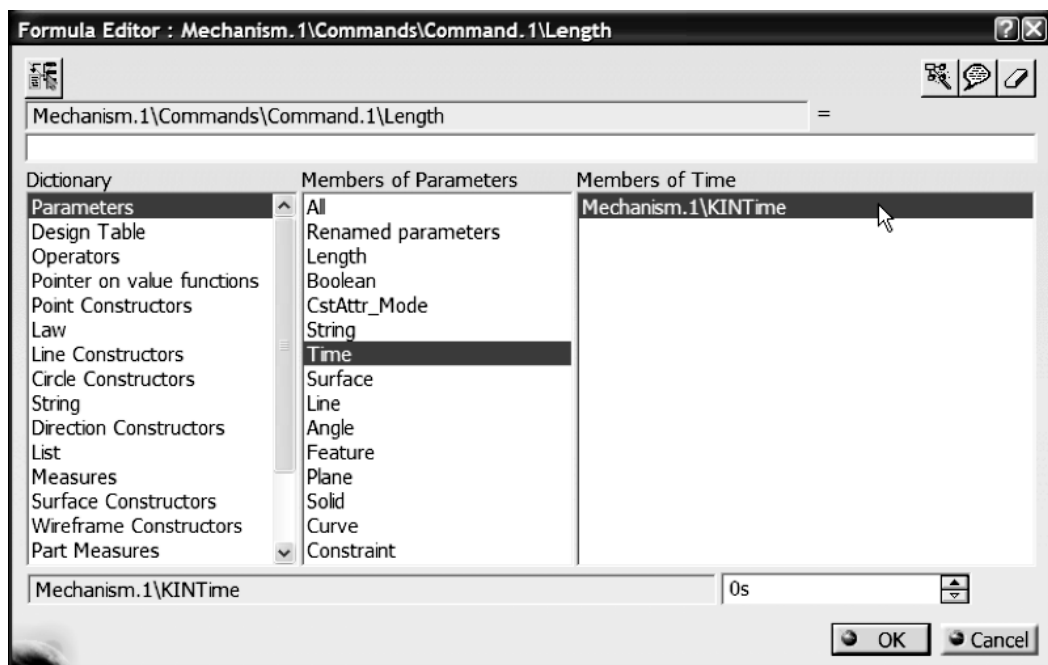


Rysunek 5.5. Widok zawartości okna *Formula Editor*

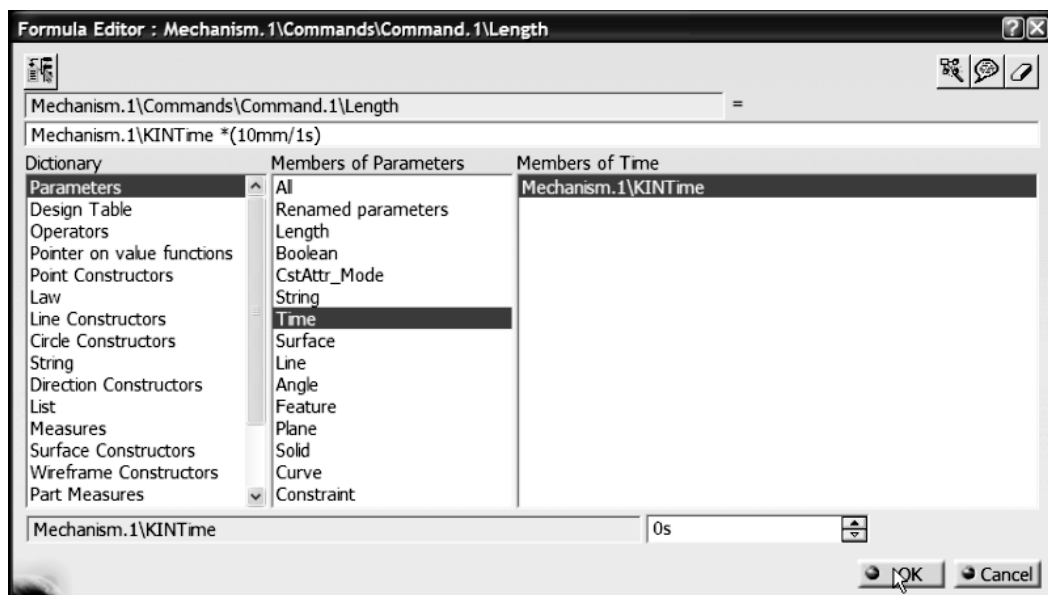
Jak już wspomniano, czas realizacji symulacji zawrzemy w parametrze *KINTime*. Aby go zastosować, należy odnaleźć go w liście *Members of Parameters* (przy aktywnej liście *Parameters*). Znajdziemy go, wskazując parametr *Time*. Wtedy to w liście *Members of Time* ukaże się poszukiwany parametr: *Mechanism.1\KINTime* (rysunek 5.6).

Za pomocą podwójnego kliknięcia na nazwie parametru dokonujemy jego skopiowania do pola edycyjnego, w którym edytujemy prawą stronę formuły (rysunek 5.7).

Następnie, aby zdefiniować prędkość ruchu elementu, zapisujemy zależność, jak na rysunku 5.7. Zapis (*10mm/1s*) oznacza, że w czasie jednej sekundy element przemieści się o 10 mm. Proces definiowania formuły kończymy, wybierając przycisk *OK*.



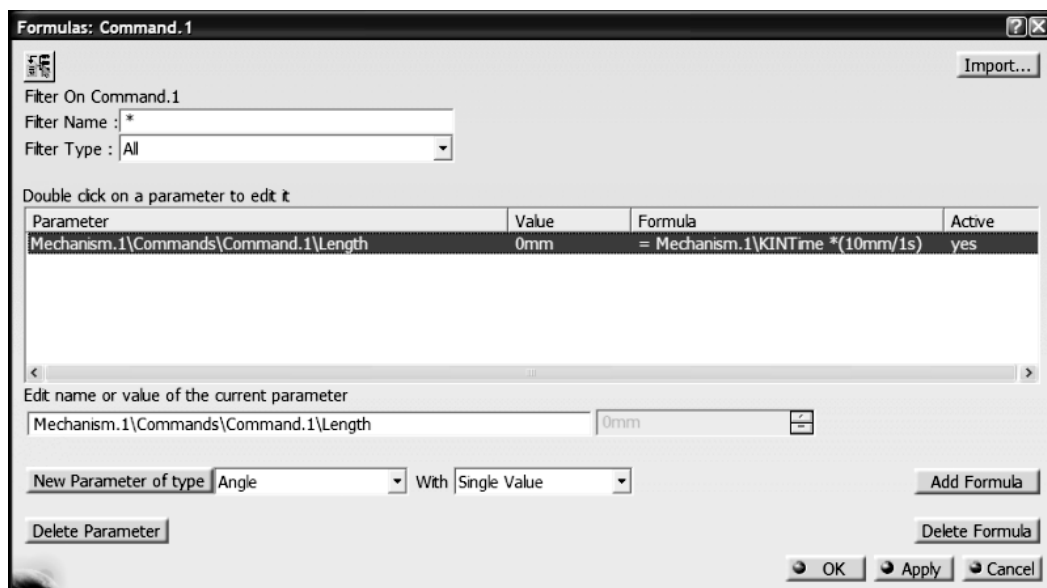
Rysunek 5.6. Widoczny parametr *Mechanism.1\KINTime*



Rysunek 5.7. Widok zdefiniowanej formuły matematycznej

W ten sposób zdefiniowaliśmy formułę, za pomocą której będzie można realizować symulację ruchu mechanizmu z pełnym nadzorem nad jego parametrami.

Po zdefiniowaniu formuły system automatycznie wraca do okna dialogowego *Formulas*. Widzimy, że wartość parametru *Mechanism.1\Commands\Command.1\Length* jest częścią zdefiniowanej formuły (rysunek 5.8). Formuła jest też domyślnie aktywna (pole *Active*).



Rysunek 5.8. Widok utworzonej formuły w oknie *Formulas*

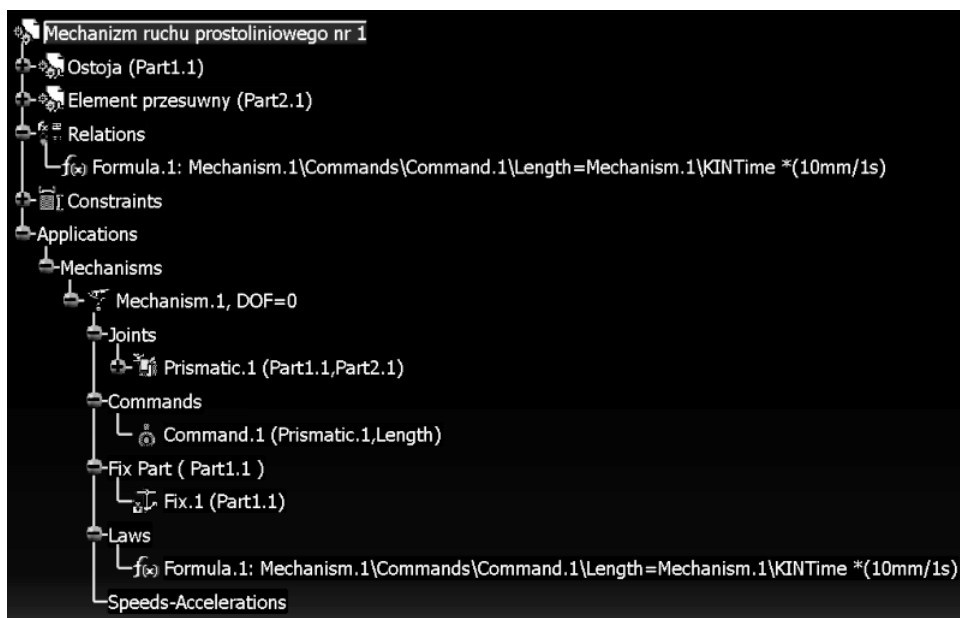
Wybraniem przycisku *Apply*, a następnie *OK* dezaktywujemy okno *Formulas*.

Utworzona formuła jest częścią całego projektu, a więc powinna być widoczna jako osobna gałąź w drzewie topologicznym. Tak też jest w rzeczywistości (rysunek 5.9). Zdefiniowana formuła znalazła się aż w dwóch miejscach drzewa topologicznego: jako podgałąź gałęzi *Relations* oraz jako podgałąź gałęzi *Laws*.

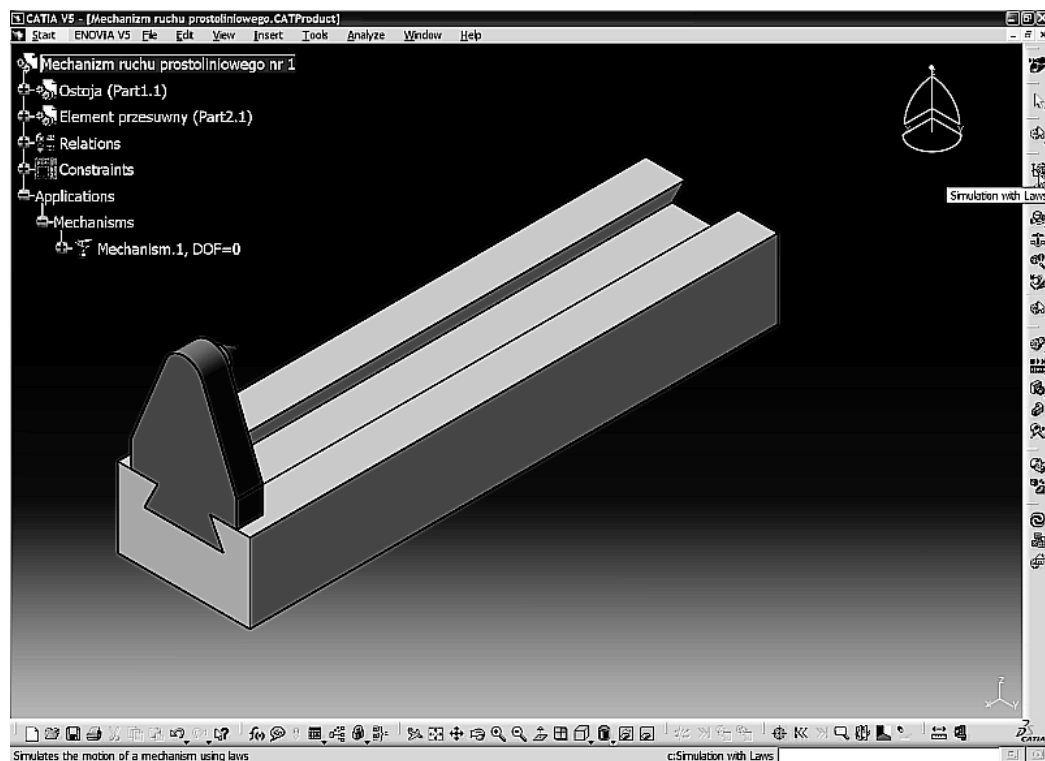
Uważny Czytelnik powinien zadać sobie pytanie: w zdefiniowanej formule zastosowaliśmy parametr *KINTime*, w którym zawarty jest czas trwania symulacji. Ale skąd wiemy, jaka jest jego wartość?

Pytanie jak najbardziej słuszne, szczególnie że wcześniej autor wspomniał o tym, że to użytkownik wpływa na długość trwania czasu symulacji. Rozwiązanie zagadki daje nam uaktywnienie polecenia *Simulation with Laws* (rysunek 5.10).

Wynikiem uaktywnienia polecenia *Simulation with Laws* jest okno *Kinematics Simulation*, zawierające przede wszystkim — znany nam już — „odtwarzacz” (rysunek 5.11). Za pomocą przycisku oznaczonego wielokropkiem (...) uruchamiamy okno *Simulation Duration*, gdzie w polu *Maximum time bound* wpisujemy czas trwania symulacji (rysunek 5.12). To właśnie tę wartość przejmuje parametr *KINTime*.



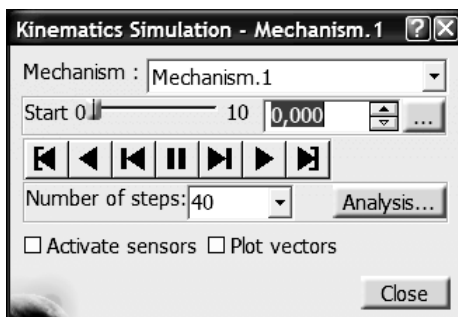
Rysunek 5.9. Widok zawartości drzewa topologicznego (widoczna utworzona formuła)



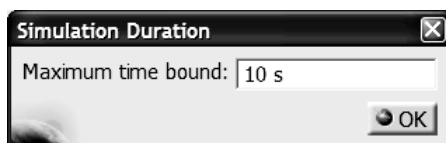
Rysunek 5.10. Uaktywnienie polecenia Simulation with Laws

Rysunek 5.11.

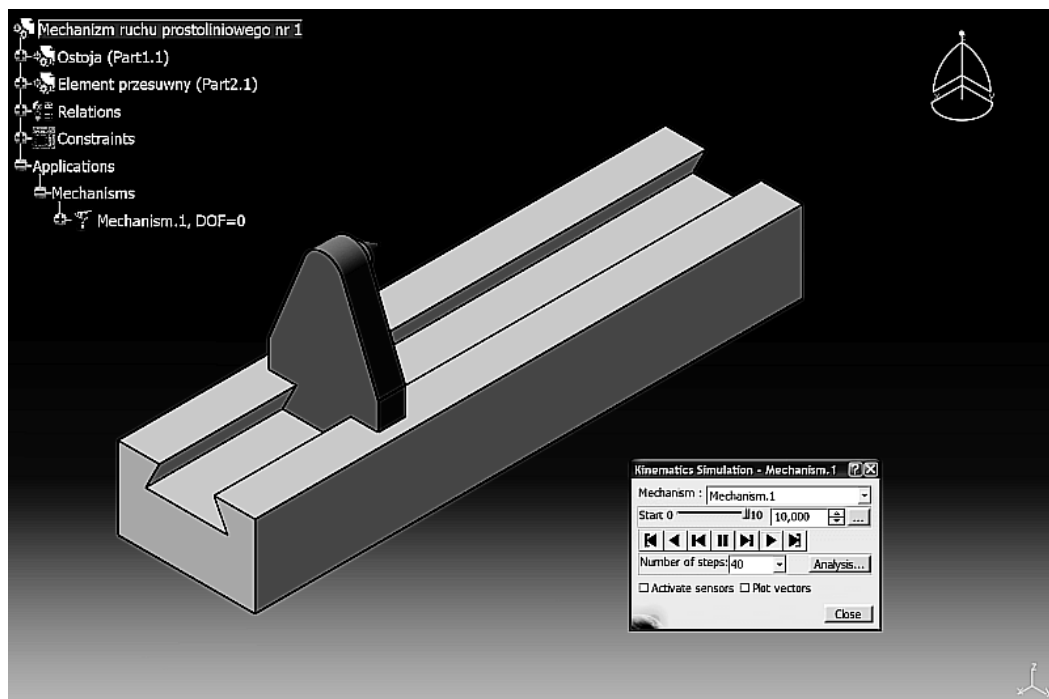
Widok zawartości
okna Kinematics
Simulation

**Rysunek 5.12.**

Widok zawartości
okna Simulation
Duration



Po uruchomieniu procesu symulacji ruchu mechanizmu *Element przesuwny* wykonuje stosowny ruch (rysunek 5.13). Porusza się on z prędkością 10 mm/s w czasie 10 s. A więc przebywa drogę 100 mm. Taką też wartość przyjmuje ostatecznie parametr *Mechanism.1\Commands\Command.1\Length*.



Rysunek 5.13. Widok położenia Elementu przesuwnego po wykonaniu symulacji ruchu

Zajmijmy się jeszcze opcjami okna *Kinematics Simulation*. Znajduje się tam m.in. opcja *Activate sensors*. Uaktywnijmy ją.

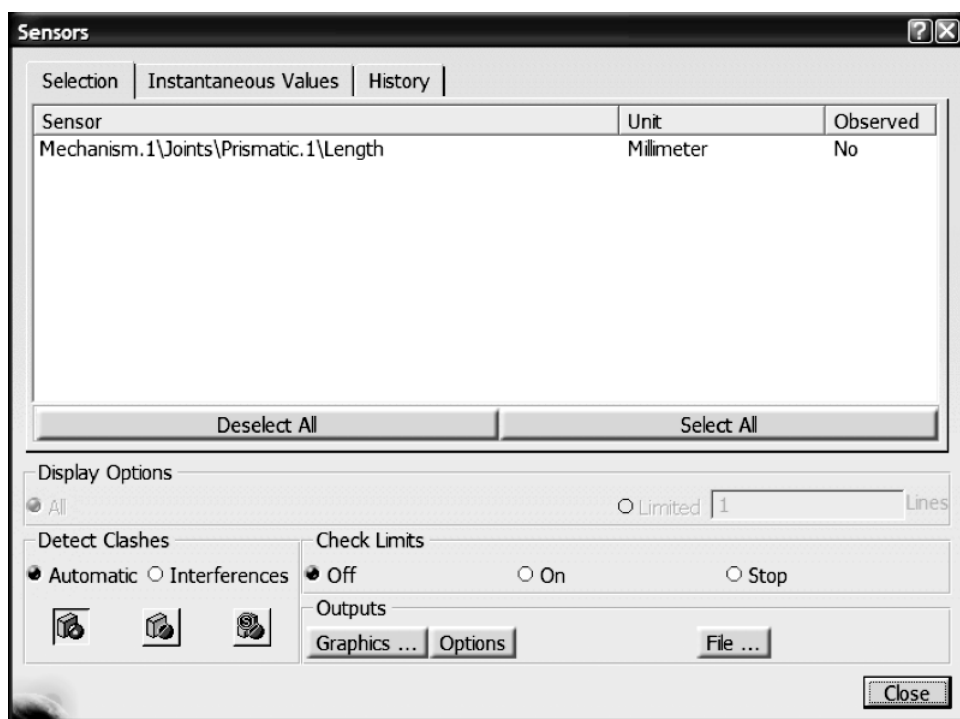


Przed wybraniem opcji *Activate sensors* należy ustawić suwak „odtwarzacza” w pozycji wyjściowej (można też dezaktywować okno *Kinematics Simulation*, a następnie uruchomić je ponownie).

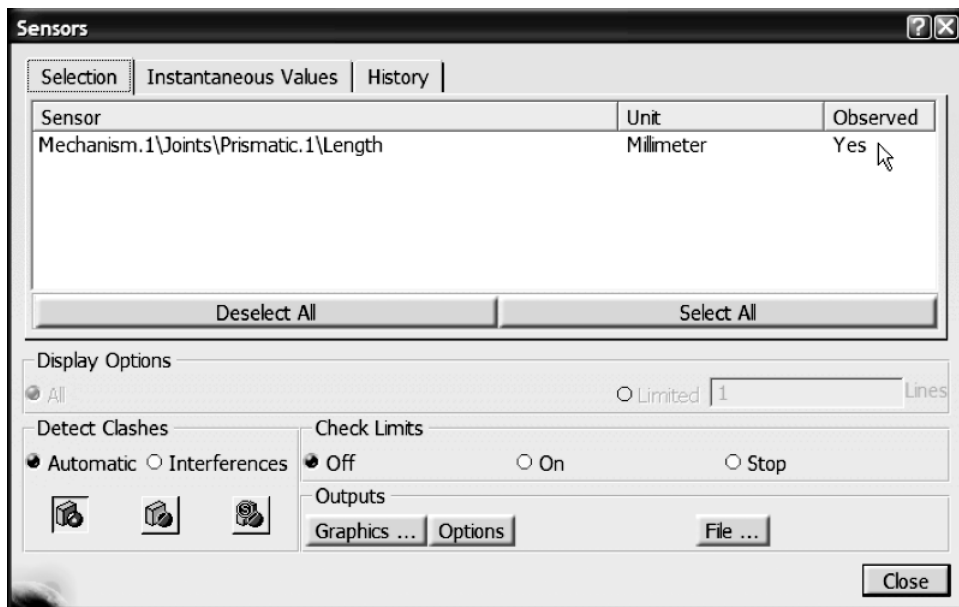
Rezultatem jej wybrania jest uaktywnienie się okna *Sensors*. Służy ono do realizacji analizy niektórych parametrów wykonanego ruchu (lub ruchów — w innych przypadkach). Bardziej szczegółowo problematyka analiz została opisana w rozdziale 6., ale — niejako przy okazji — omówmy kilka podstawowych zagadnień.

Otóż okno to posiada trzy zakładki: *Selection*, *Instantaneous Values* oraz *History*. Zajmijmy się pierwszą z nich.

Zawartością tej zakładki (rysunek 5.14) jest pole zawierające wszystkie utworzone więzy ruchowe (w naszym przypadku tylko jeden), których wartości przemieszczeń są uzależnione od zastosowanej formuły i tym samym ulegają zmianie podczas realizacji procesu symulacji. Przebieg zmienności ich wartości w funkcji czasu możemy obserwować graficznie. Aby to uczynić, najpierw musimy wskazać systemowi, że chcemy obserwować przebieg zmienności danej wartości w czasie. W tym celu należy zmienić wartość zmiennej w kolumnie *Observed* z *No* (rysunek 5.14) na *Yes* (rysunek 5.15).

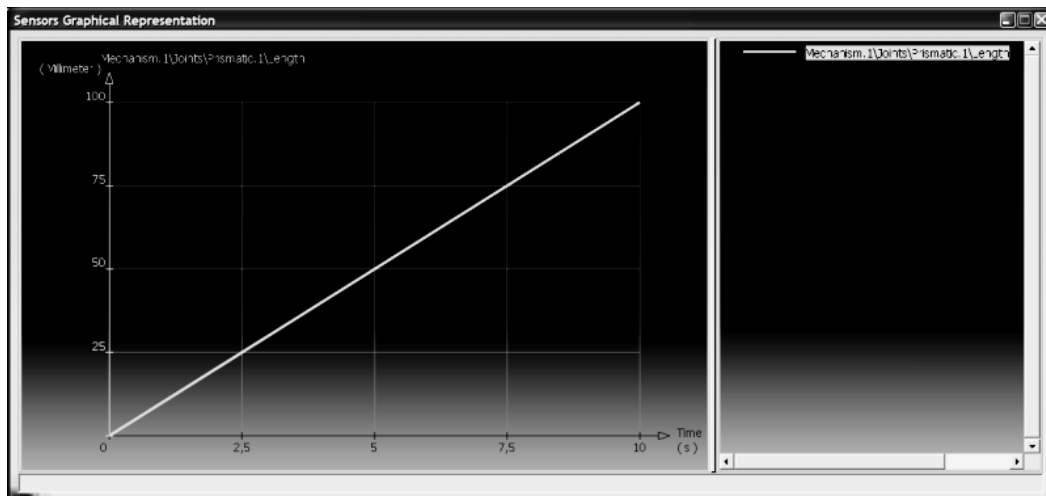


Rysunek 5.14. Widok zawartości zakładki *Selection* okna *Sensors*



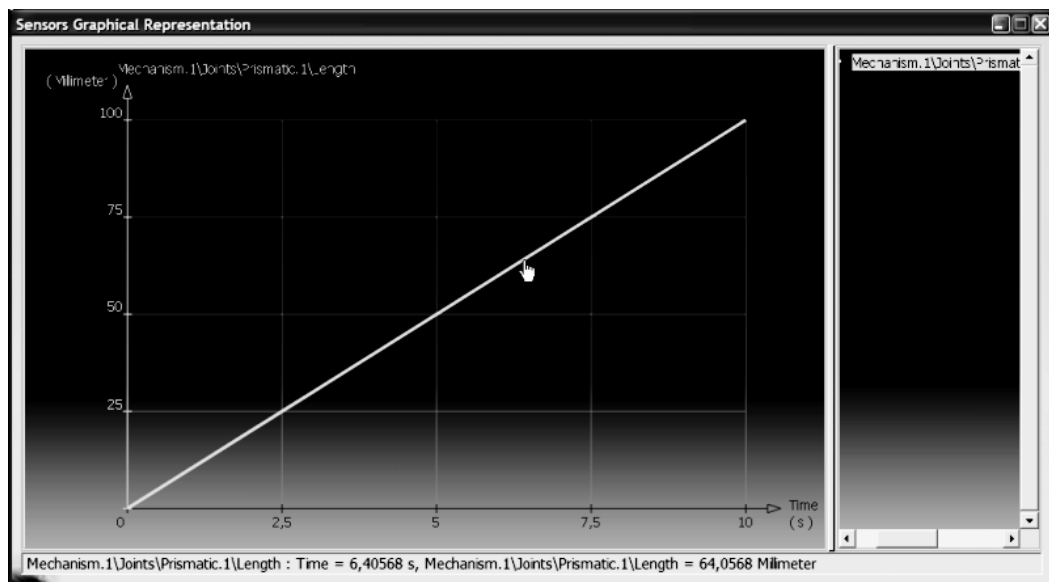
Rysunek 5.15. Uaktywnienie zmiennej aktywności obserwowanej wielkości

Po wykonaniu symulacji ruchu wraz z uaktywnieniem obserwowania zmienności wartości przemieszczenia w więzie *Mechanism.1\Joints\Prismatic.1\Length* wybieramy przycisk *Graphics* (z okna *Sensors*). Uaktywnia się wtedy okno *Sensors Graphical Representation* przedstawiające w sposób graficzny (za pomocą wykresu liniowego) przebieg zmian wartości wybranej wielkości w funkcji czasu (rysunek 5.16). Jak widzimy, model przebył dystans 100 mm w czasie 10 s ze stałą prędkością.



Rysunek 5.16. Widok zawartości okna *Sensors Graphical Representation*

Widoczna linia wykresu jest aktywna. Po ustawieniu w dowolnym jej miejscu kursora myszki w dolnej części okna ukazują się wartości czasu (*Time*) i odległości (*Length*), odpowiadające wskazanemu punktowi (rysunek 5.17).



Rysunek 5.17. Widok wartości czasu i odległości odpowiadających wskazanemu punktowi wykresu prędkości

Sposób odczytu wartości poprzez wskazanie kursorem myszy jest oczywiście niedokładny i trudny do praktycznego zastosowania. Dlatego też istnieje dodatkowo możliwość eksportu danych związanych z otrzymanym wykresem do postaci arkusza kalkulacyjnego MS Excel.

Eksportu dokonujemy, klikając — znajdujący się w dolnej części okna *Sensors* — przycisk *File*. Rezultatem jest uaktywnienie się okna *Save As* (rysunek 5.18), za pomocą którego możemy dokonać stosownego zapisu otrzymanych danych do pliku *.xls* o dowolnej nazwie.

Po otwarciu pliku z użyciem programu MS Excel widzimy wartości wygenerowanych danych uporządkowanych w trzech kolumnach (rysunek 5.19).

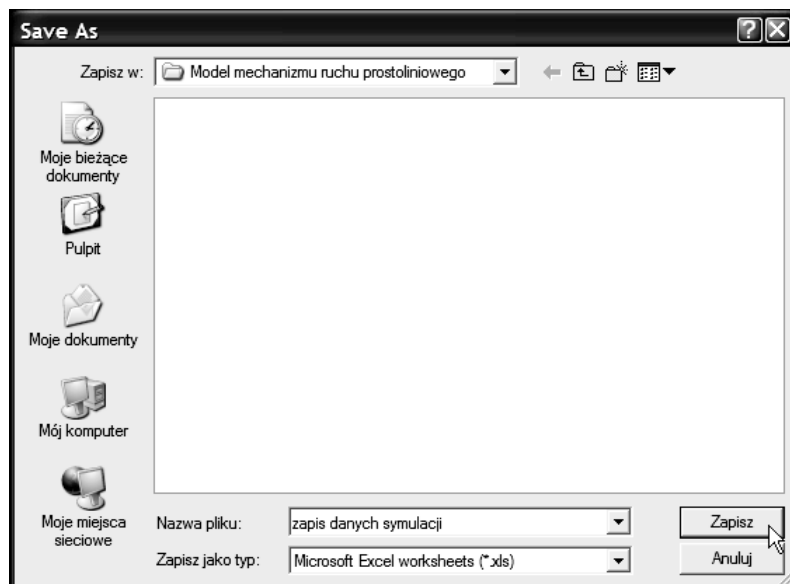


Wartości w kolumnach *B* oraz *C* są identyczne. W kolumnie *B* liczby są traktowane jako wartości komend w poszczególnych krokach czasowych, natomiast w kolumnie *C* są one traktowane jako wartości parametru przemieszczenia.

Przejdźmy teraz do zakładki *Instantaneous Values* (rysunek 5.20). Widzimy tam listę podzieloną na trzy kolumny, podobnie jak w zakładce *Selection*. Na liście tej możemy obserwować chwilowe wartości przemieszczeń w węźle *Mechanism.1\Joints\Prismatic.1\Length*. W przypadku, jaki widzimy na rysunku 5.20, jest to wartość końcowa.

Rysunek 5.18.

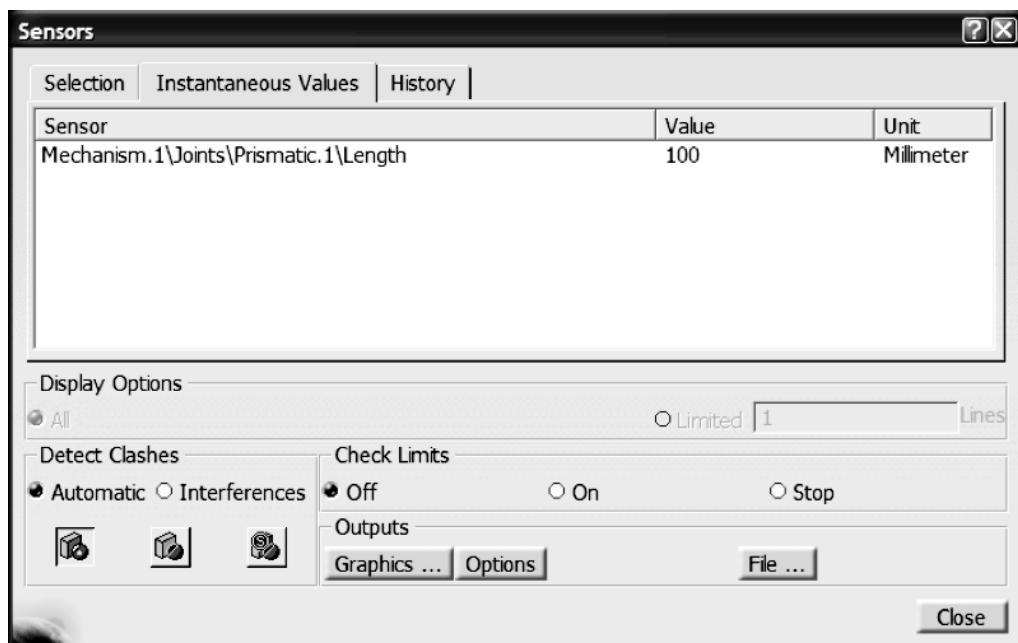
Okno zapisu danych
z wykonanej
symulacji ruchu
do postaci pliku
arkusza kalkulacyjnego
MS Excel

**Rysunek 5.19.**

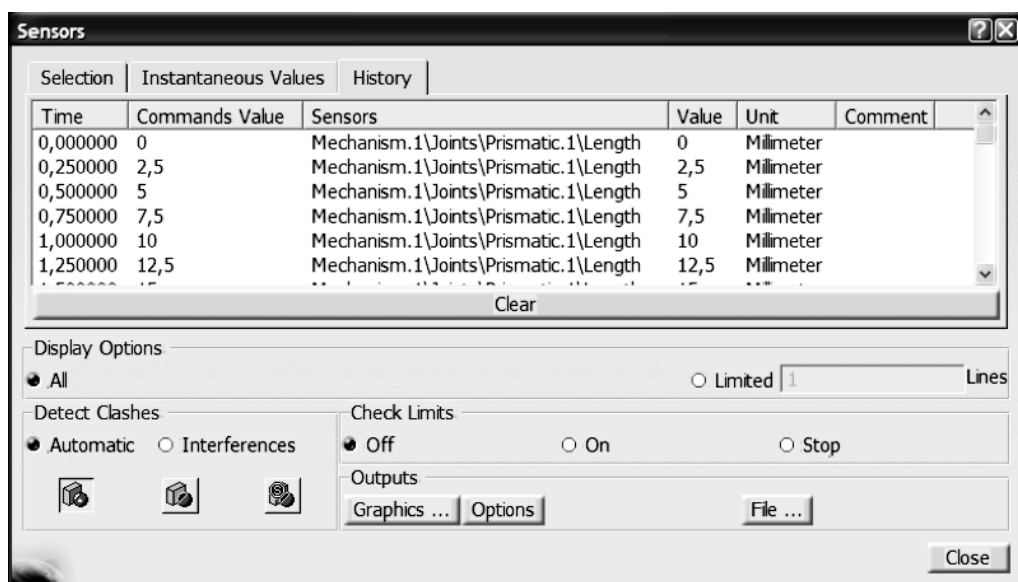
Widok fragmentu
zawartości arkusza
kalkulacyjnego
programu MS Excel

	A	B	C
1	Time	Commands Value	Mechanism.1\Joints\Prismatic.1\Length(Millimeter)
2	0	0,00	0
3	0,25	2,50	2,5
4	0,5	5,00	5
5	0,75	7,5	7,5
6	1	10,00	10
7	1,25	12,5	12,5
8	1,5	15,00	15
9	1,75	17,5	17,5
10	2	20,00	20
11	2,25	22,5	22,5
12	2,5	25,00	25
13	2,75	27,5	27,5
14	3	30,00	30
15	3,25	32,5	32,5
16	3,5	35,00	35
17	3,75	37,5	37,5
18	4	40,00	40
19	4,25	42,5	42,5
20	4,5	45,00	45
21	4,75	47,5	47,5
22	5	50,00	50

Na zakończenie przeglądania okna *Sensors* przejdźmy do zakładki *History* (rysunek 5.21). Widzimy tam znów listę o większej liczbie kolumn. Możemy tam obserwować wszystkie wartości, jakie podczas realizacji symulacji przybierał parametr *Mechanism.1\Commands\Command.1\Length*. Jest to swego rodzaju odpowiednik arkusza kalkulacyjnego, do którego podobne dane można zapisać. Zawartość listy zakładki *History* można w dowolnym momencie usunąć za pomocą przycisku *Clear*.



Rysunek 5.20. Widok zawartości zakładki *Instantaneous Values*

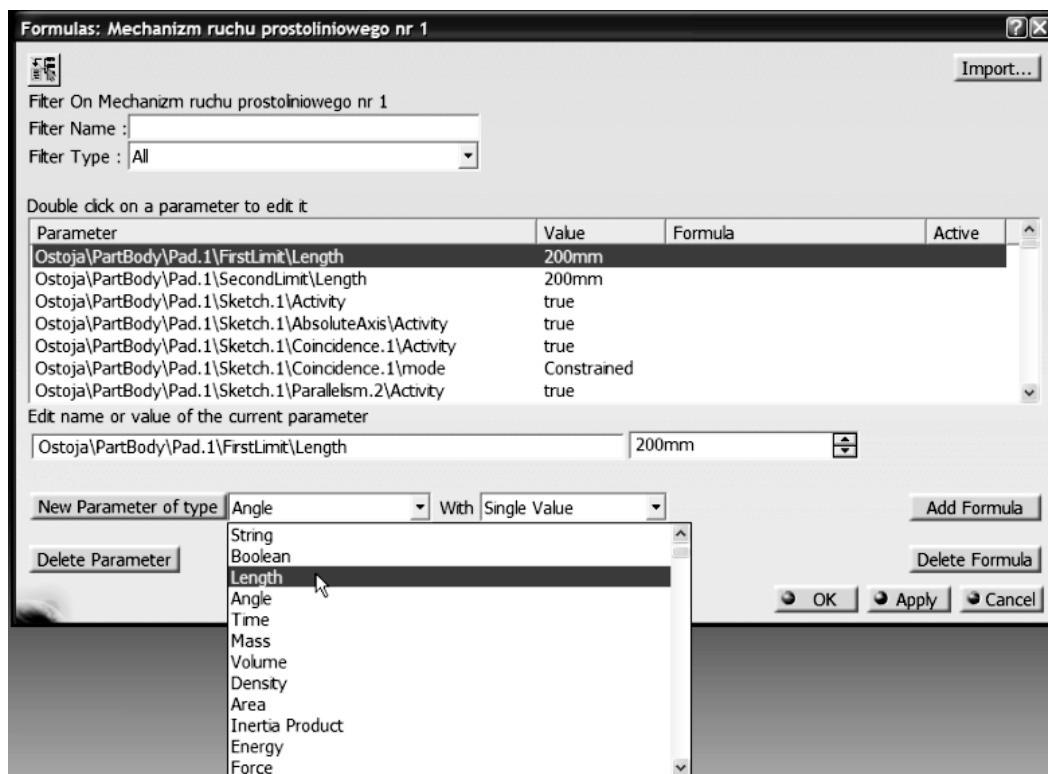


Rysunek 5.21. Widok zawartości zakładki *History* okna *Sensors*

Jak wspomniano we wstępie do niniejszego rozdziału, *CATIA* jest systemem parametrycznym. A więc spróbujmy wykorzystać tę własność systemu do usprawnienia realizacji symulacji z użyciem komend.

W przypadku gdybyśmy chcieli np. zwiększyć prędkość przesuwu elementu wykonującego ruch, wiązałoby się to z wykonaniem edycji zapisanej formuły. Taki sposób postępowania jest możliwy, ale niewygodny. Można więc sobie wyobrazić sytuację, gdy w zapisie formuły wystąpi zmienna, której wartości będziemy zmieniać poza „ciałem” formuły. Natomiast wszelkie zmiany jej wartości zostaną automatycznie przesłane do formuły.

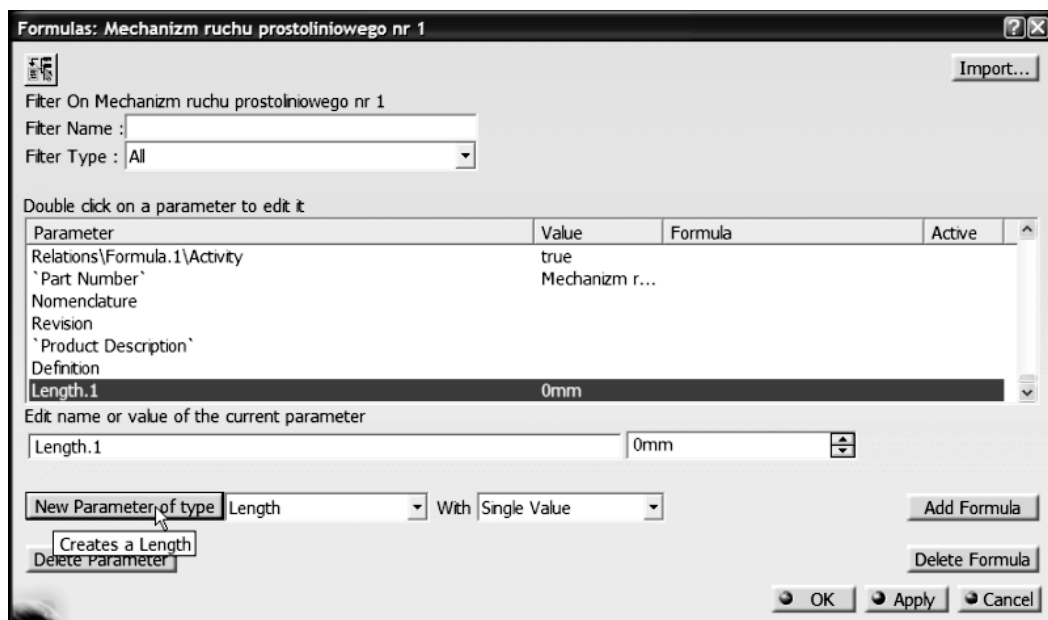
Właśnie do zrealizowania takiego sposobu postępowania wykorzystamy parametryczność systemu. W pierwszym kroku musimy utworzyć nową zmienną, typu *Length* (rysunek 5.22). W tym celu ponownie uaktywniamy okno *Formulas*. Z listy rozwijalnej, jak na rysunku 5.22, wybieramy typ zmiennej — *Length*. Następnie klikamy przycisk *New Parameter of type* (rysunek 5.23).



Rysunek 5.22. Wybór typu nowej zmiennej

Nowa zmienna zostaje utworzona. Zmieńmy jej nazwę z — domyślnej — *Length* na *droga* (rysunek 5.24).

Skutkiem utworzenia nowej zmiennej jest również jej pojawienie się w strukturze drzewa topologicznego (rysunek 5.25). Zauważmy, że domyślną wartością zmiennej *droga* jest *0mm*.



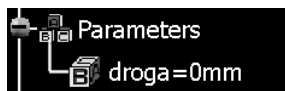
Rysunek 5.23. Utworzenie nowej zmiennej typu *Length*



Rysunek 5.24. Zmiana nazwy zmiennej z *Length* na *droga*

Rysunek 5.25.

Widok fragmentu
drzewa topologicznego
(widoczna gałąź
nowego parametru)

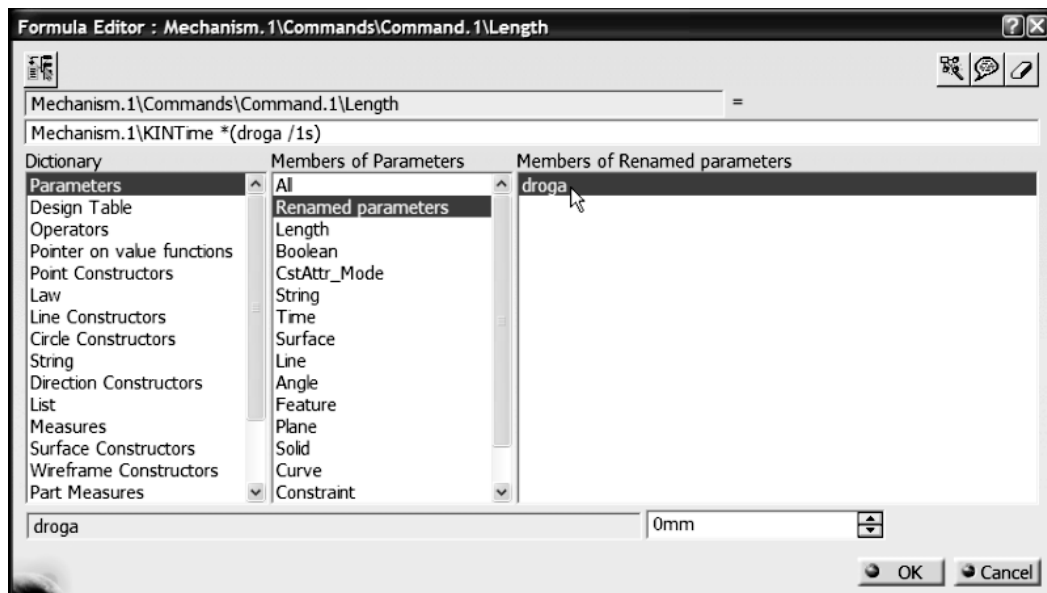


Aby zastosować nową zmienną w — poprzednio utworzonej — formule, należy przejść do jej edycji (np. wykonując klik podwójny myszką na jej gałęzi w drzewie topologicznym). W miejsce stałej wartości *100mm* wstawmy nazwę utworzonej zmiennej (rysunek 5.26). Nazwę utworzonego parametru można szybko znaleźć w liście *Members of Parameters* w pozycji *Renamed parameters* (rysunek 5.26).

Zmianę treści formuły zatwierdzamy poprzez wybór przycisku *OK*.

Zmiana treści formuły natychmiast widoczna jest również w drzewie topologicznym (rysunek 5.27).

Aby móc efektywnie użyć nowego parametru do symulacji ruchu, musimy zmienić jego wartość. W tym celu wykonujemy klik podwójny lewym przyciskiem myszy na nazwie parametru w drzewie topologicznym. Efektem tego jest pojawienie się okna edycyjnego *Edit Parameter* (rysunek 5.28).



Rysunek 5.26. Użycie zmiennej droga w formule



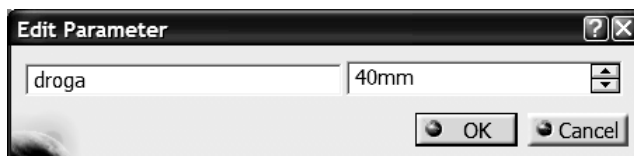
Rysunek 5.27. Widok zmienionej zawartości gałęzi formuły Formula.1

Rysunek 5.28.
Widok zawartości
okna Edit Parameter



Zmieniamy domyślną wartość zmiennej *droga* z *0mm* na np. *40mm* (rysunek 5.29).

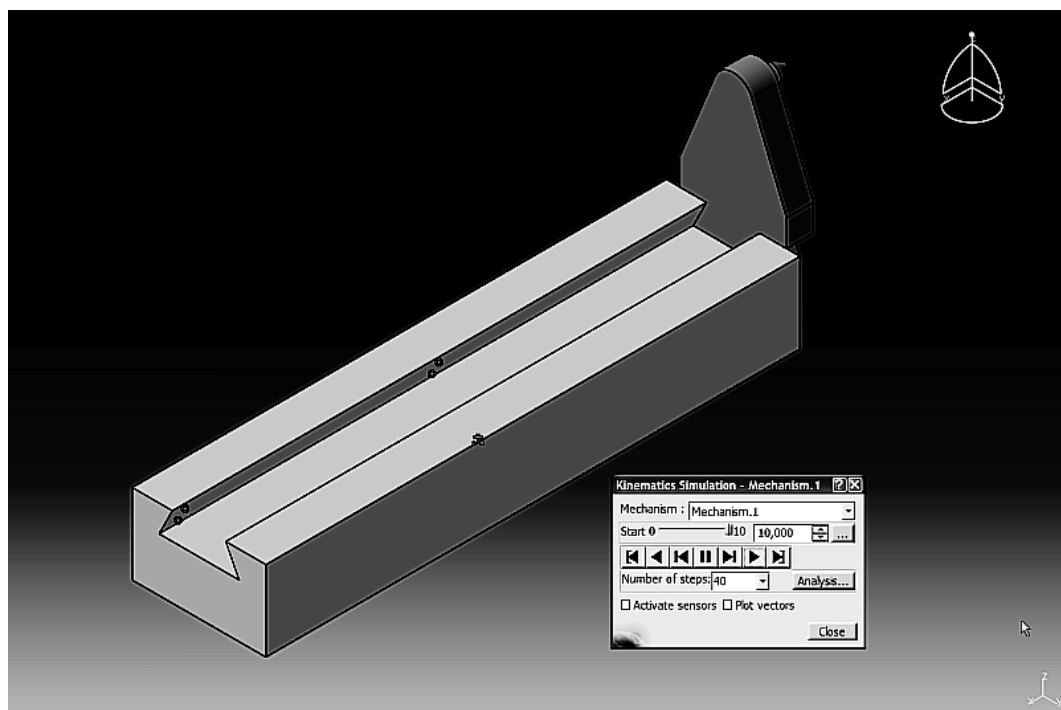
Rysunek 5.29.
Widok zawartości
okna Edit Parameter
— widoczna nowa
wartość zmiennej
droga



Możemy teraz odtworzyć symulację ruchu mechanizmu z użyciem polecenia *Simulation with Laws*. Widzimy, że *Element przesuwny* w czasie 10 s pokonał drogę równą długości prowadnicy (*Ostoi*), czyli 400 mm (rysunek 5.30).



Chcąc zmieniać wartość prędkości *Elementu przesuwnego*, wystarczy każdorazowo zmieniać wartość parametru *droga*, nie ingerując już w treść formuły.



Rysunek 5.30. Widok położenia Elementu przesuwnego po wykonaniu symulacji ruchu

Zastosowanie formuł do symulacji ruchu obrotowego

Po zapoznaniu się ze sposobem zastosowania formuł do sterowania realizacją ruchu jednostajnego prostoliniowego zajmiemy się zastosowaniem formuł do sterowania ruchem również jednostajnym, ale obrotowym.

Dla lepszej ilustracji problemu posłużymy się przykładem. W tym celu, otwierając katalog *Modele/Rozdział 5/Model mechanizmu ruchu obrotowego* (z załączonej płyty CD-ROM), wskazujemy plik *Mechanizm ruchu obrotowego.CATProduct*.

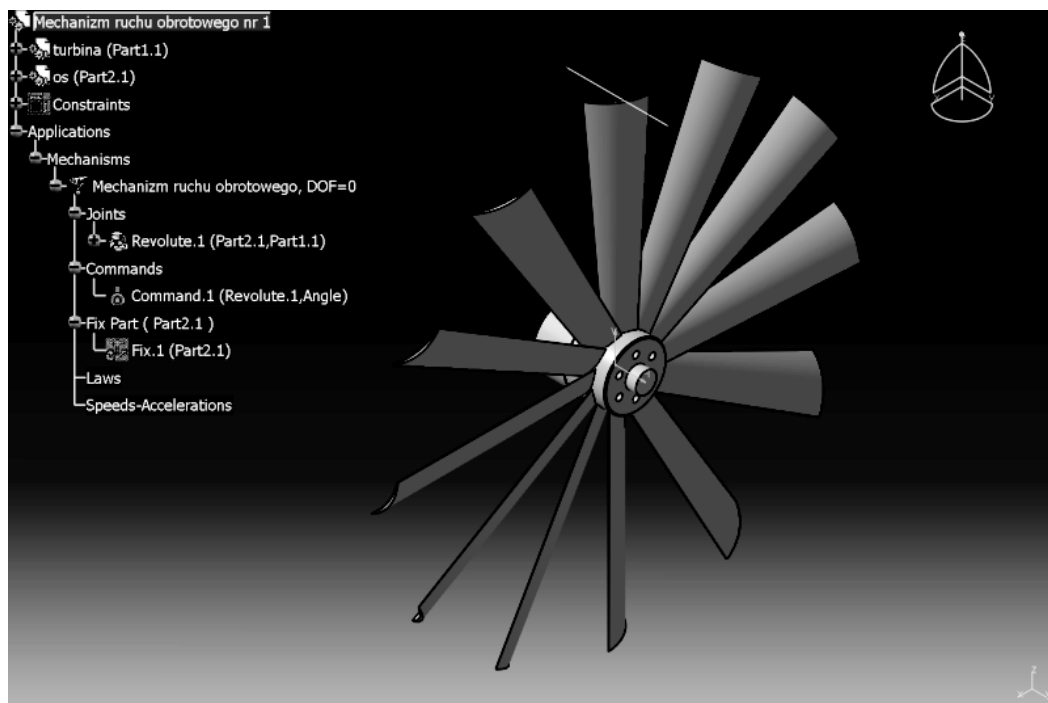
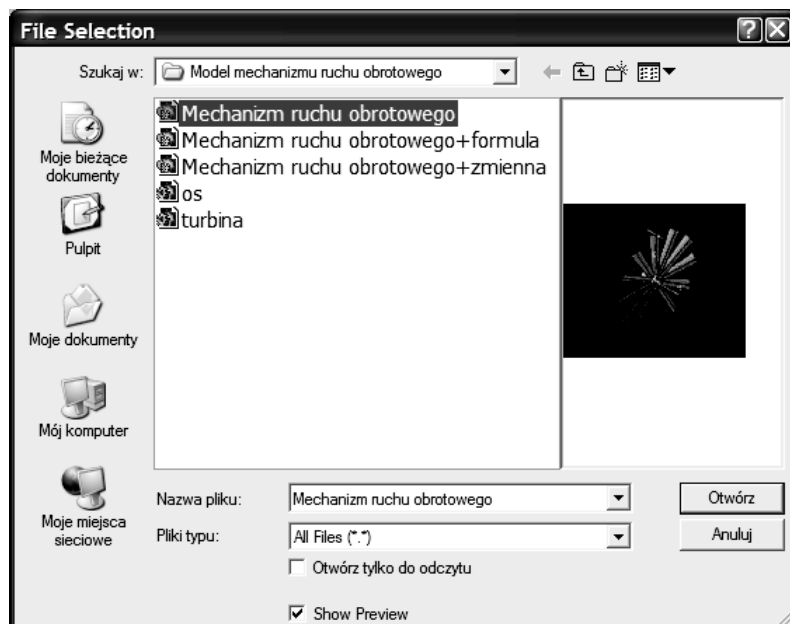
W oknie podglądu okna *File Selection* widzimy postać modelu mechanizmu (rysunek 5.31).

Działanie mechanizmu ruchu obrotowego jest bardzo proste: model turbiny obraca się wokół osi. Dla lepszego zauważenia efektu ruchu do modelu załączono dwa odcinki (w położeniu wyjściowym nakładają się na siebie, rysunek 5.32). Jeden odcinek związany jest z modelem turbiny, natomiast drugi został związany z modelem osi.

Cechą charakterystyczną ruchu obrotowego jest zmiana kąтового położenia elementu obracającego się w jednostce czasu. Prędkość kątową takiego ruchu opisujemy za pomocą zależności: **prędkość kątowa = kąt obrotu/czas**.

Rysunek 5.31.

Widok zawartości okna
File Selection podczas
otwierania pliku
Mechanizm
ruchu obrotowego.
CATProduct

**Rysunek 5.32.** Widok postaci mechanizmu ruchu obrotowego

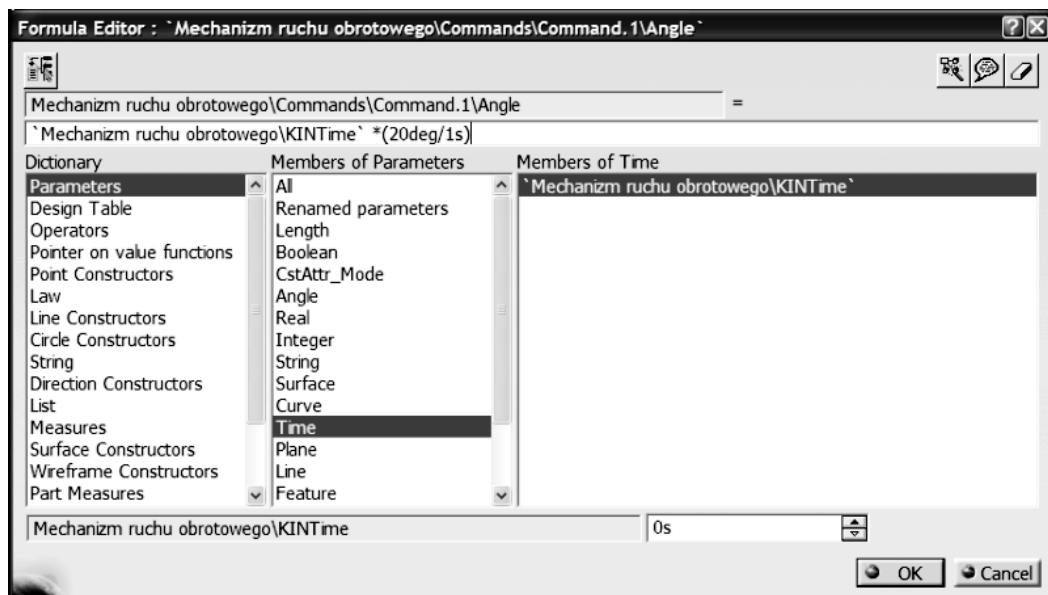
W przypadku naszego modelu mechanizmu mamy zdefiniowany jeden więz sterowalny, gdzie sterujemy kątem obrotu (więz typu *Revolute*). Analogicznie więc do poprzedniego przykładu musimy przekształcić wzór opisujący prędkość kątową na zależność opisującą kąt obrotu. Po stosownym przekształceniu otrzymujemy zależność: **kąt obrotu = czas × prędkość kątowa**.

Postępując analogicznie jak w poprzednim przykładzie, definiujemy formułę, opisującą ruch obrotowy modelu turbiny. Prawa strona formuły tym razem będzie zawierała, oprócz parametru czasu (*KINTime*), prędkość kątową.



Notację jednostki stopnia kąowego (s) w systemie *CATIA* zastąpiono skrótem *deg*.

Z analizy treści zapisanej formuły z rysunku 5.33 wynika, że model turbiny w czasie 1 s dokona obrotu o 20°, a więc będzie się poruszał z prędkością kątową 20°/1 s.



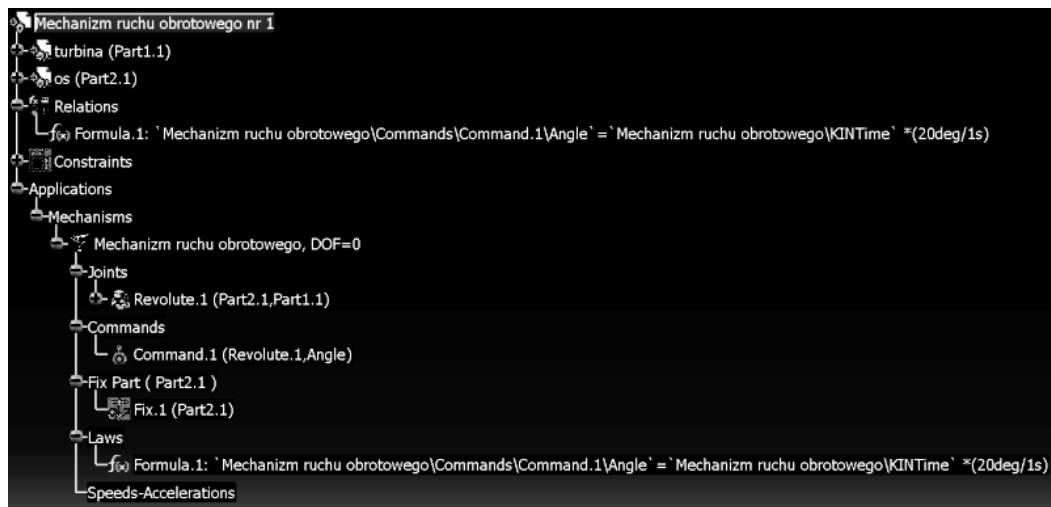
Rysunek 5.33. Zapis formuły sterującej ruchem obrotowym — okno *Formula Editor*



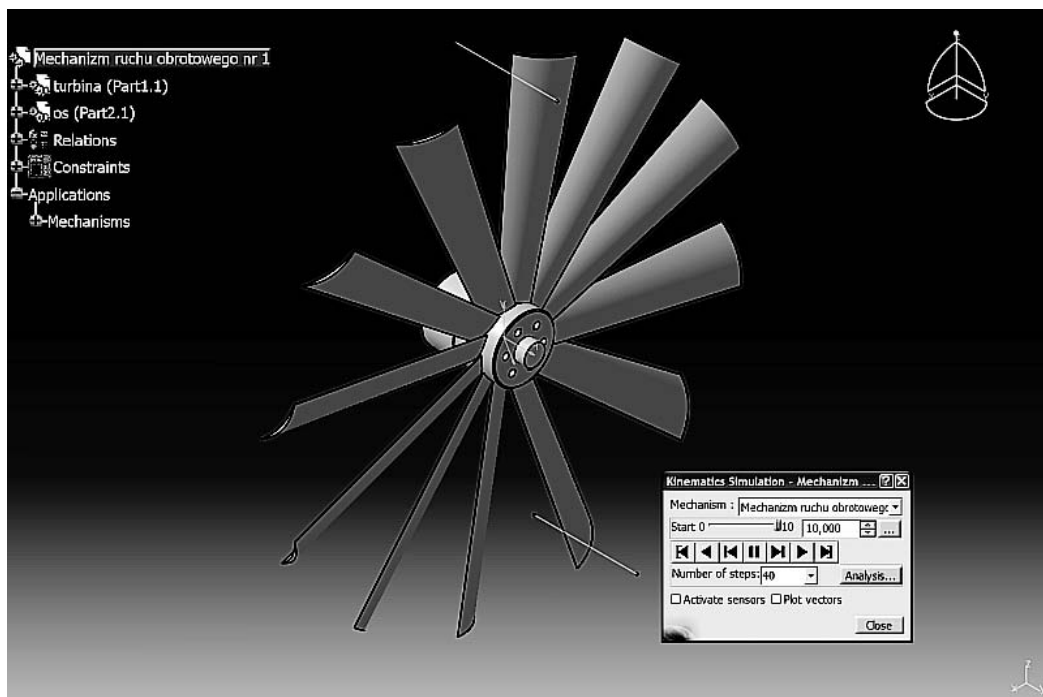
Domyślną nazwę mechanizmu zmieniono na: *Mechanizm ruchu obrotowego*.

Zdefiniowana formuła jest również widoczna w strukturze drzewa topologicznego (rysunek 5.34).

Mając zdefiniowaną formułę, sterującą realizacją ruchu obrotowego, możemy wykonać symulację ruchu mechanizmu. W tym celu korzystamy z polecenia *Simulation with Laws* (rysunek 5.35). Czas realizacji ruchu ustawiamy na 10 s.



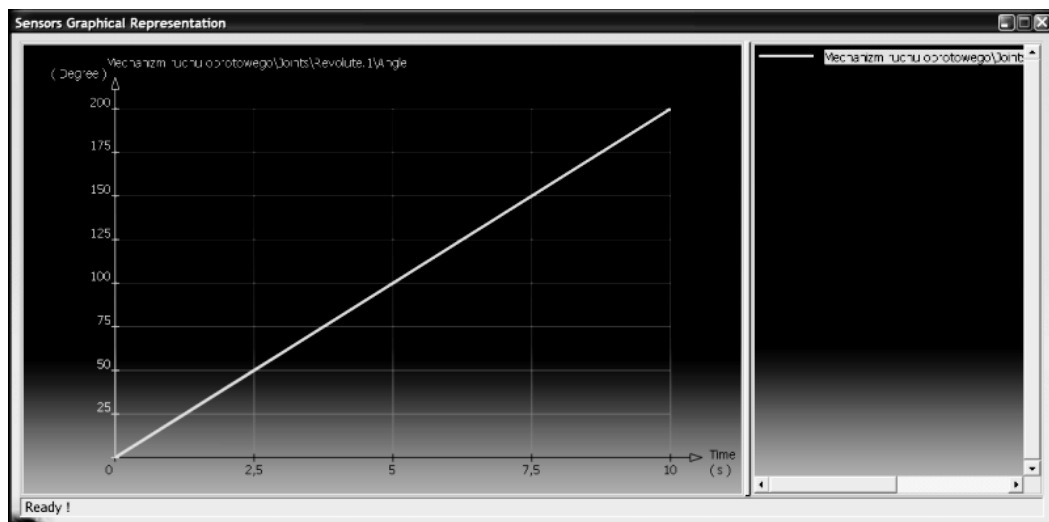
Rysunek 5.34. Widok drzewa topologicznego modelu mechanizmu — widoczne gałęzie zawierające formułę Formula.1



Rysunek 5.35. Widoczny efekt symulacji ruchu

Widzimy, że model turbiny w czasie 10 s przemieścił się o 200° (widoczne kątowe „rozejście” się odcinków kontrolnych, rysunek 5.35).

Korzystając z opcji *Activate sensors* okna *Kinematics Simulation*, widzimy w postaci graficznej przebieg realizacji ruchu (rysunek 5.36). Na wykresie widzimy linię prostą, charakterystyczną dla ruchu jednostajnego.



Rysunek 5.36. Widok wykresu stałej prędkości kątowej



Gdyby skale na obu osiach wykresu miały identyczny wymiar, linia wykresu byłaby pochylona pod kątem 45° .

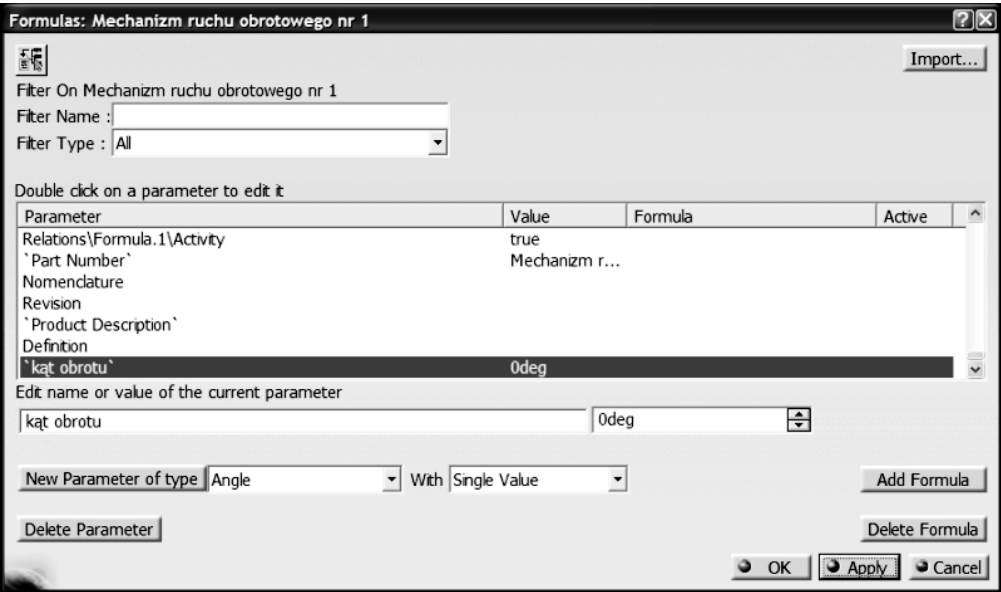
Podobnie jak w poprzednim przykładzie, zdefiniujemy dodatkową zmienną, aby móc „wygodnie” sterować wartością prędkości kątowej. Korzystając z polecenia *Formulas*, tworzymy nowy parametr, ale tym razem typu *Angle*, nadając mu nazwę *kąt obrotu* (rysunek 5.37).

Utworzony parametr widzimy również w strukturze drzewa topologicznego. Jego domyślna wartość to 0° (rysunek 5.38).

Zmienimy więc domyślną wartość, aby prędkość kątowa była różna od zera. Do zmiany wartości zmiennej *kąt obrotu* użyjemy — tym razem — polecenia *kąt obrotu object/Definition* z menu kontekstowego prawego przycisku myszy (rysunek 5.39). Efektem wyboru tego polecenia jest uaktywnienie się okna edycyjnego *Edit Parameter* (rysunek 5.40). Zmieniamy wartość na 10deg .

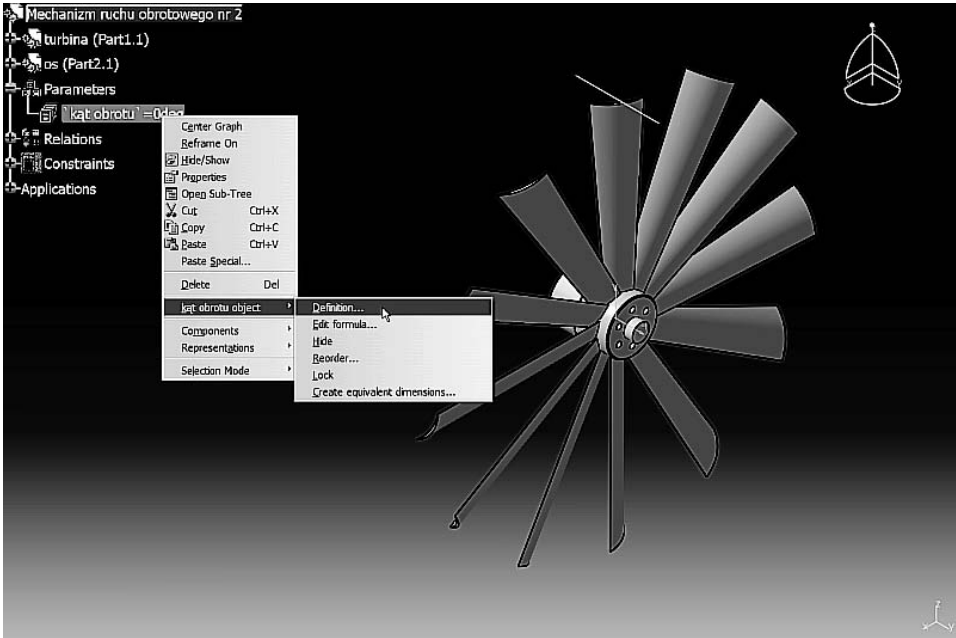
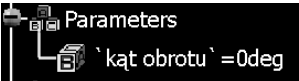
Po utworzeniu nowego parametru i ustaleniu jego wartości należy dokonać stosownej zmiany w zapisie treści formuły (rysunek 5.41).

Po ponownym wykonaniu symulacji ruchu widzimy, że model turbiny przemieścił się w ciągu 10 s o kąt 100° (rysunek 4.42). A więc prędkość kątowa wynosiła $10^\circ/\text{s}$.



Rysunek 5.37. Utworzenie nowego parametru typu Angle

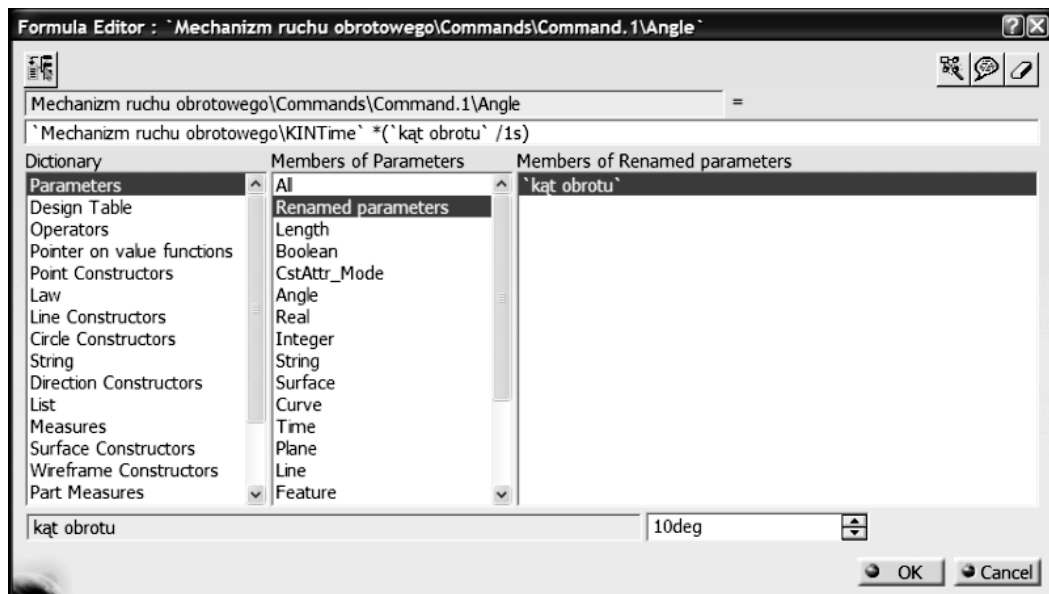
Rysunek 5.38.
Widok domyślnej wartości
parametru kąt obrotu



Rysunek 5.39. Zastosowanie menu kontekstowego prawego przycisku myszy do zmiany wartości parametru kąt obrotu

Rysunek 5.40.

Zmiana wartości
parametru *kąt obrotu*

Rysunek 5.41. Zmiana treści formuły z użyciem parametru *kąt obrotu*

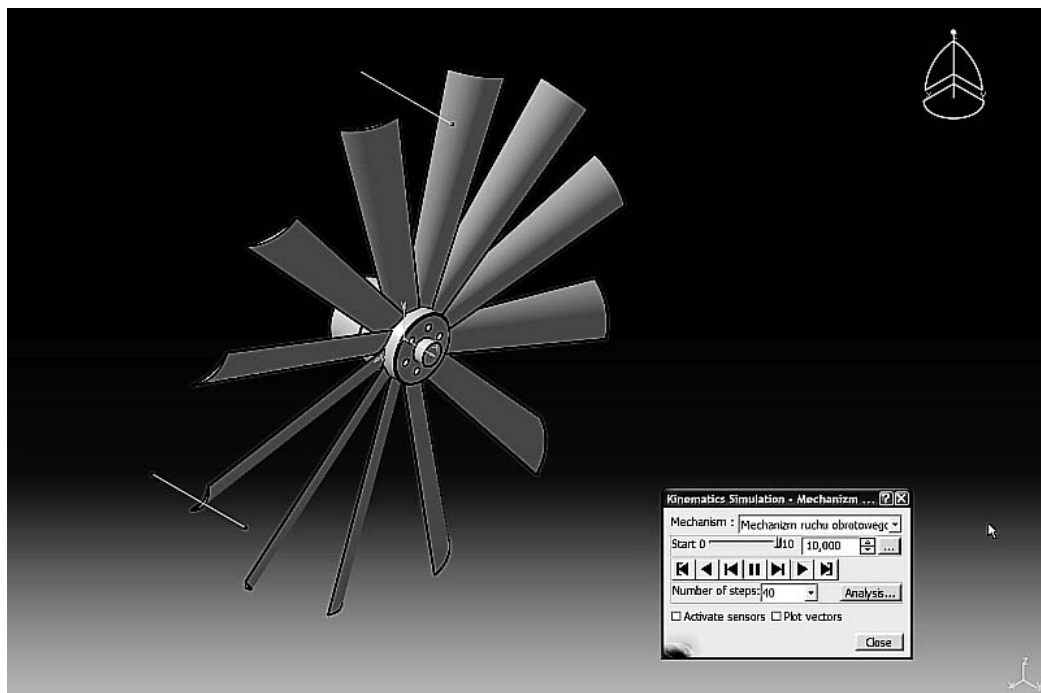
Zastosowanie formuł do symulacji ruchu złożonego

Kolejny przykład jest — swego rodzaju — podsumowaniem obu poprzednich przykładów. Dotychczas poznaliśmy sposób definiowania formuł służących do sterowania symulacją ruchów: liniowego oraz — osobno — obrotowego. W przykładzie niniejszym zapoznamy się ze sterowaniem za pomocą formuł ruchu złożonego. Będzie to ruch złożony z ruchu obrotowego i liniowego.

Jako przykład posłuży nam uproszczony **model procesu wiercenia otworu**. W tym celu, otwierając katalog *Modele/Rozdział 5/Symulacja wiercenia otworu* (z załączonej płyty CD-ROM), wskazujemy plik *Wiercenie otworu.CATProduct*.

W oknie podglądu okna *File Selection* widzimy postać modelu mechanizmu (rysunek 5.43).

Charakterystyczną cechą kinematyki procesu wiercenia otworów jest ruch złożony narzędzia — wiertła. Wiertło wykonuje ruch obrotowy oraz ruch postępowy. Mamy więc do czynienia z synchronizacją dwóch rodzajów ruchów. Przy niezmiennych warunkach procesu wiercenia można więc założyć stałość ruchu roboczego wiertła w danym przedziale czasu.



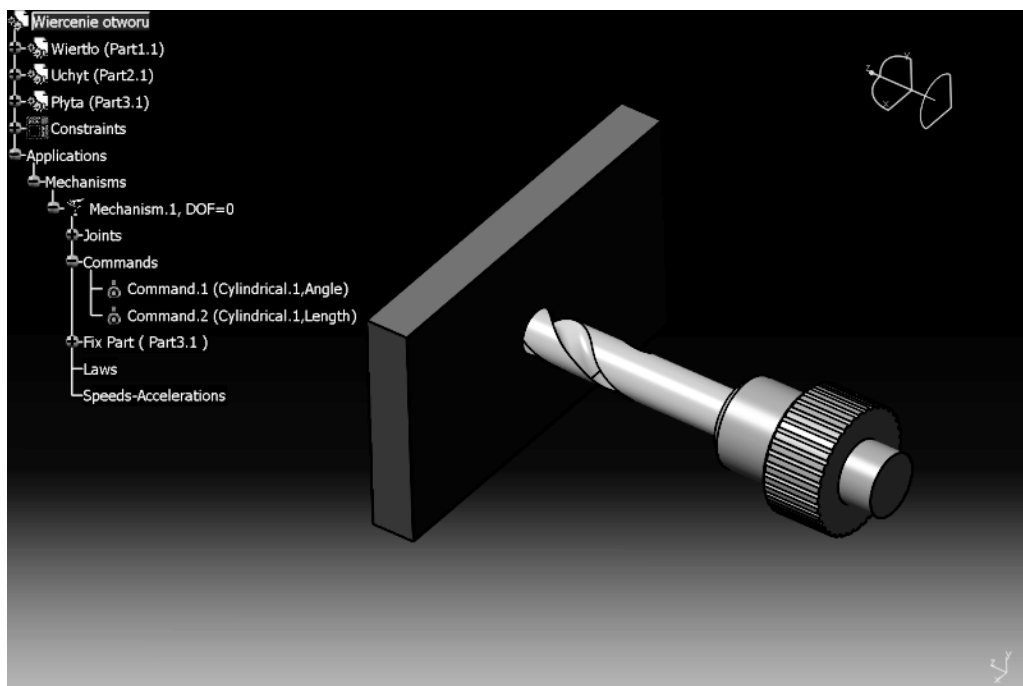
Rysunek 5.42. Widok efektu symulacji ruchu

Rysunek 5.43.

Widok zawartości okna *File Selection* podczas otwierania pliku *Wiercenie otworu.CATProduct*



Realizację takiego rodzaju ruchu złożonego umożliwia więc kinematyczny *Cylindrical Joint*. Został on zastosowany do sterowania ruchem modelu wiertła (rysunek 5.44). W drzewie topologicznym widzimy więc dwa sterowania: kątem obrotu (*Command.1*) oraz posuwem (*Command.2*).



Rysunek 5.44. Widok postaci modelu mechanizmu wiercenia

Zastosowanie formuł do sterowania tego rodzaju ruchem złożonym będzie więc polegało na ciągłej zmianie wartości parametrów (sterowań) w danym przedziale czasu.

Posługując się wiedzą nabytą w ramach realizacji dwóch poprzednich przykładów, utworzymy dwie formuły. Pierwsza będzie związana z parametrem *Command.1*, a więc ze sterowaniem ruchem obrotowym. W tym celu dla parametru *Command.1* definiujemy formułę, jak na rysunku 5.45. Prędkość kątowna modelu wiertła wyniesie więc 60°/s.

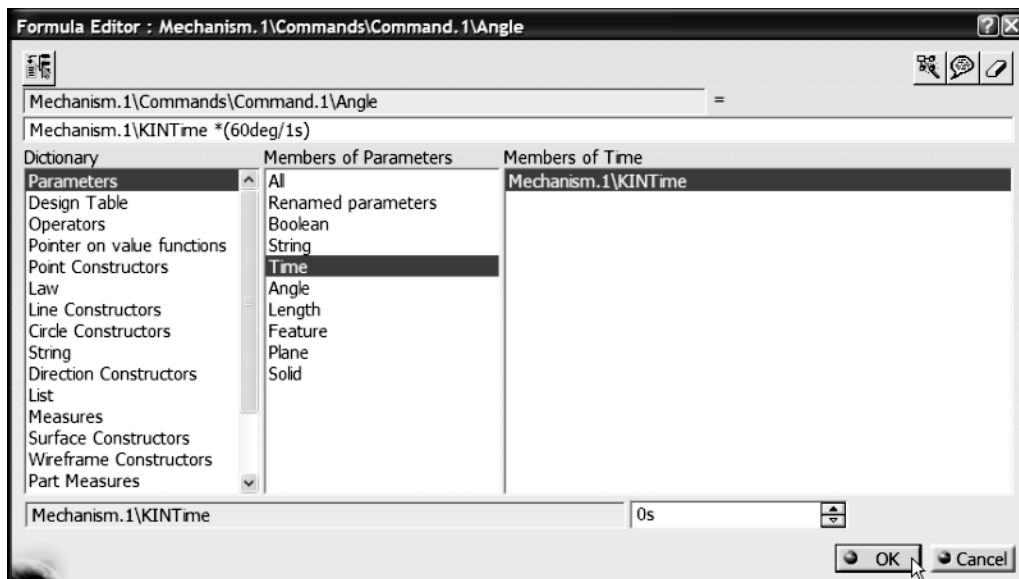


Wybór tak małej prędkości kątowej wynika wyłącznie ze specyfiki przykładu i jest całkowicie nieadekwatny do rzeczywistych prędkości procesu wiercenia.

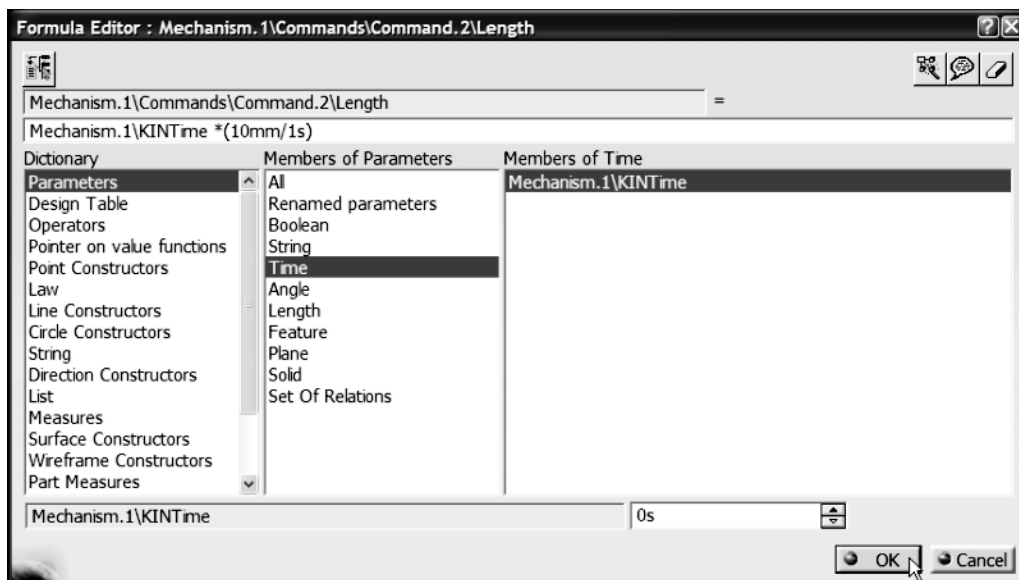
Postępując podobnie, definiujemy stosowną formułę dla parametru *Command.2* (formuła będzie sterowała ruchem posuwistym). Zakładamy więc przykładową prędkość posuwu — 10 mm/s (rysunek 5.46).

Obie utworzone formuły widoczne są w strukturze drzewa topologicznego (rysunek 5.47).

Po utworzeniu obu formuł możemy wykonać symulację ruchu z użyciem polecenia *Simulation with Laws*. Tym razem ustalamy czas trwania symulacji na 6 s (rysunek 5.48). Podczas realizacji symulacji ruchu wyraźnie widzimy synchronizację obu ruchów: obrotowego oraz posuwowego modelu wiertła.

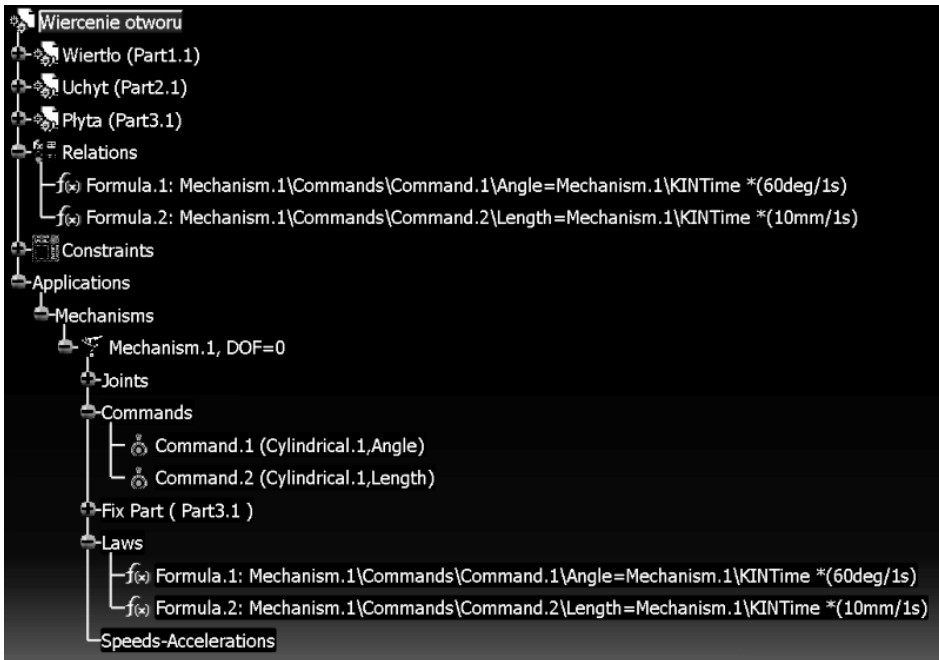


Rysunek 5.45. Widok treści formuły sterującej realizacją ruchu obrotowego

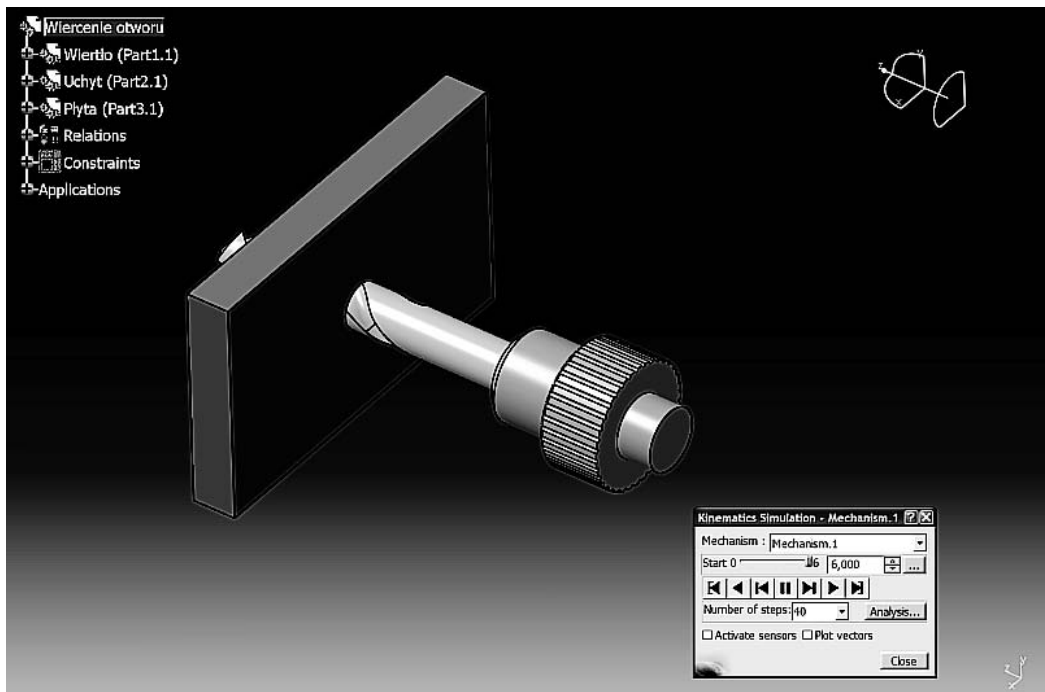


Rysunek 5.46. Widok treści formuły sterującej realizacją ruchu posuwowego

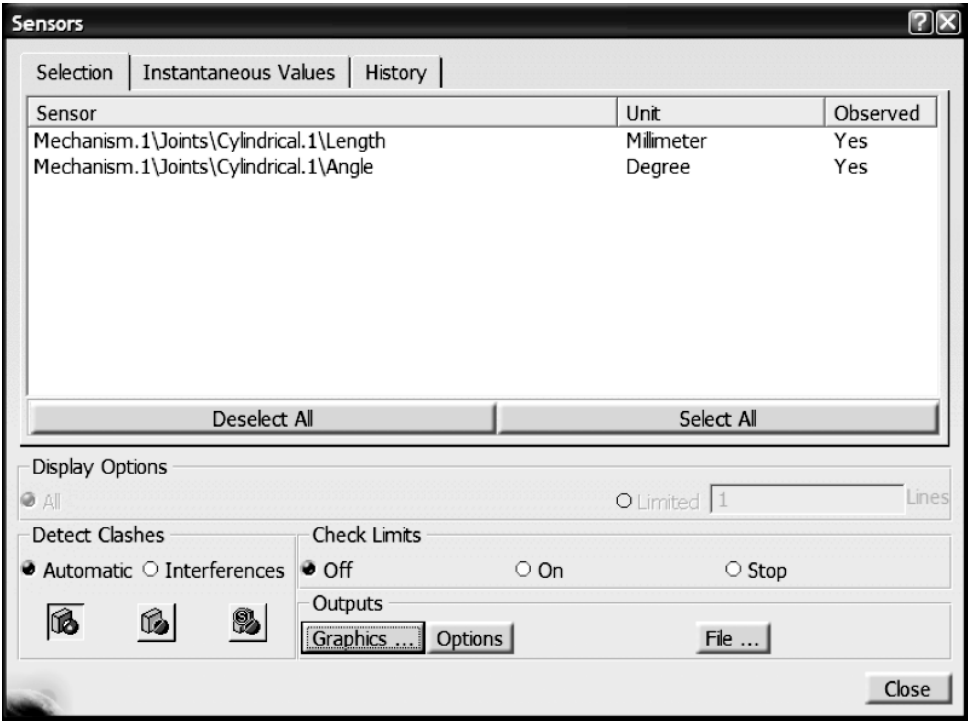
Uruchamiając opcję *Activate sensors* okna *Kinematics Simulation*, a następnie aktywując „obserwację” przebiegu zmienności obu sterowanych parametrów ruchu (rysunek 5.49), możemy zobaczyć przebieg symulacji ruchu złożonego na wykresach (rysunek 5.50). Tym razem widzimy dwa wykresy: osobny dla prędkości liniowej modelu oraz osobny dla prędkości kątowej. Obie prędkości są stałe w całym przedziale czasowym symulacji ruchu.



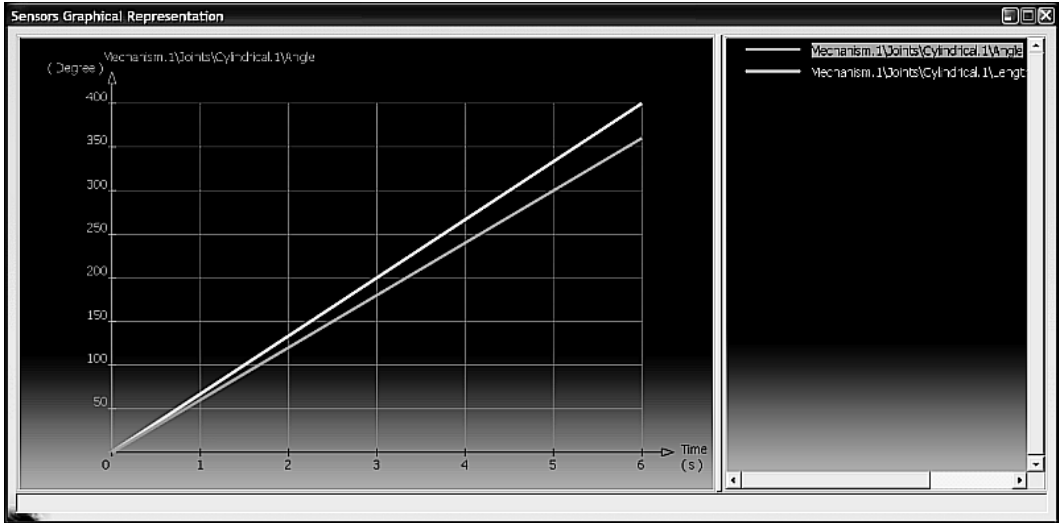
Rysunek 5.47. Widok zawartości drzewa topologicznego mechanizmu wiercenia otworu



Rysunek 5.48. Widoczny efekt realizacji symulacji ruchu (położenie modelu wiertła w końcowej chwili realizacji ruchu)



Rysunek 5.49. Aktywacja obserwacji zmian wartości sterowanych parametrów ruchu



Rysunek 5.50. Widok postaci wykresów: prędkości liniowej oraz kątowej modelu wiertła — w funkcji czasu



Klikając lewym przyciskiem myszy dany wykres, uzyskujemy możliwość odczytu wartości z nim związanych.

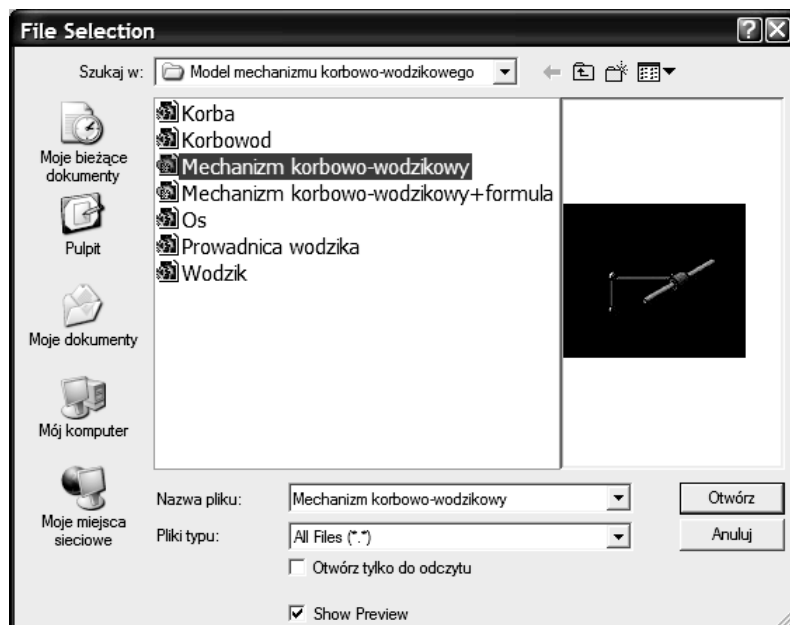
Zastosowanie formuł i funkcji matematycznych do symulacji ruchu złożonego

Ostatni przykład, związany z zastosowaniem formuł do symulacji ruchu, dotyczy wzbogacenia treści formuł o zastosowanie funkcji matematycznych (dostarczanych przez system *CATIA*). Zastosowanie wybranych funkcji matematycznych pozwala na znaczne urozmaicenie postaci wykonywanego przez mechanizm ruchu, wraz z jego matematyczną systematyzacją.

Przykład zastosowania funkcji matematycznej do budowy formuły związany jest ze sterowaniem działania modelu **mechanizmu korbowo-wodzikowego**. Stosowny plik znajdziemy w katalogu *Modele/Rozdział 5/Model mechanizmu korbowo-wodzikowego* (z załączonej płyty CD-ROM). Wybieramy plik *Mechanizm korbowo-wodzikowy.CATProduct*.

W oknie podglądu okna *File Selection* widzimy postać modelu mechanizmu (rysunek 5.51).

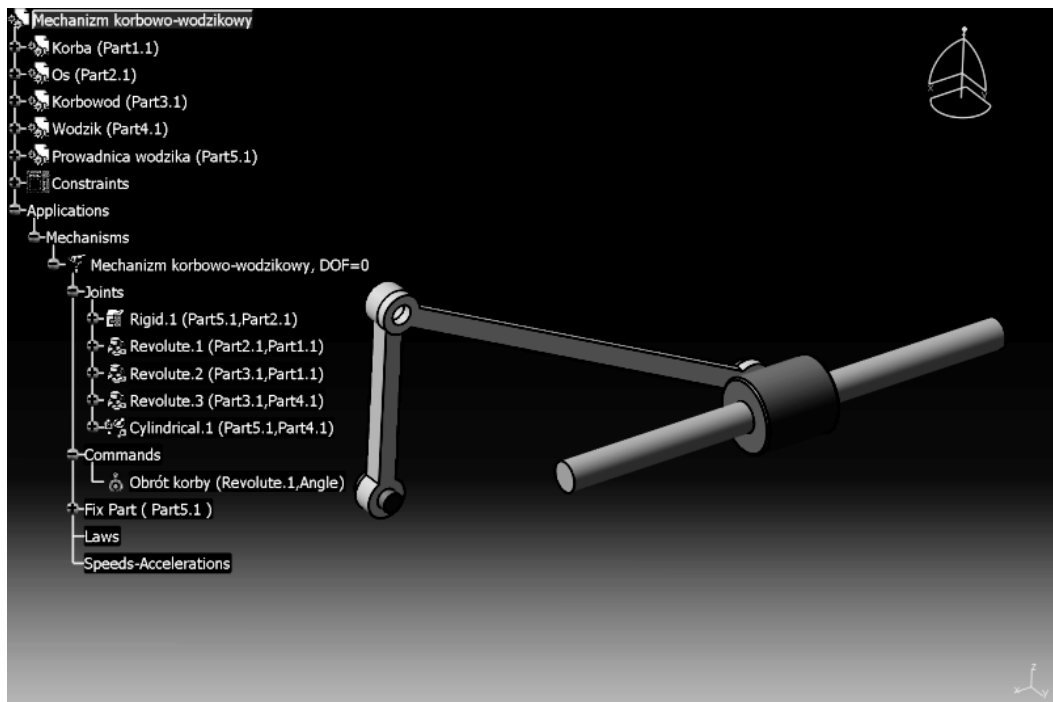
Rysunek 5.51.
Widok zawartości
okna *File Selection*
podczas otwierania
pliku *Mechanizm
korbowo-wodzikowy.CATProduct*



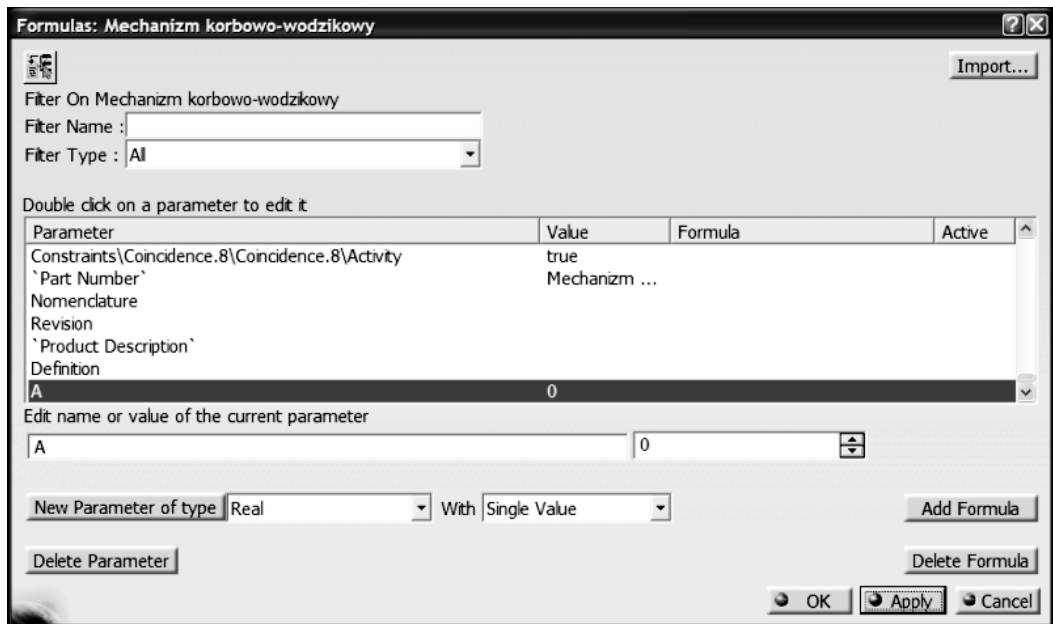
Działanie podobnego mechanizmu poznaliśmy już w ramach rozdziału 3. Tym razem jednak korba będzie wykonywała ruchy wahadłowe, o charakterze harmonicznym. Postać mechanizmu widoczna jest na rysunku 5.52.

Zanim zdefiniujemy stosowną formułę, utwórzmy dwie zmienne (zostaną one zastosowane do jej budowy). Jako pierwszą utwórzmy zmienną typu *Real* o nazwie *A* (rysunek 5.53). Będzie ona oznaczała **amplitudę** ruchu.

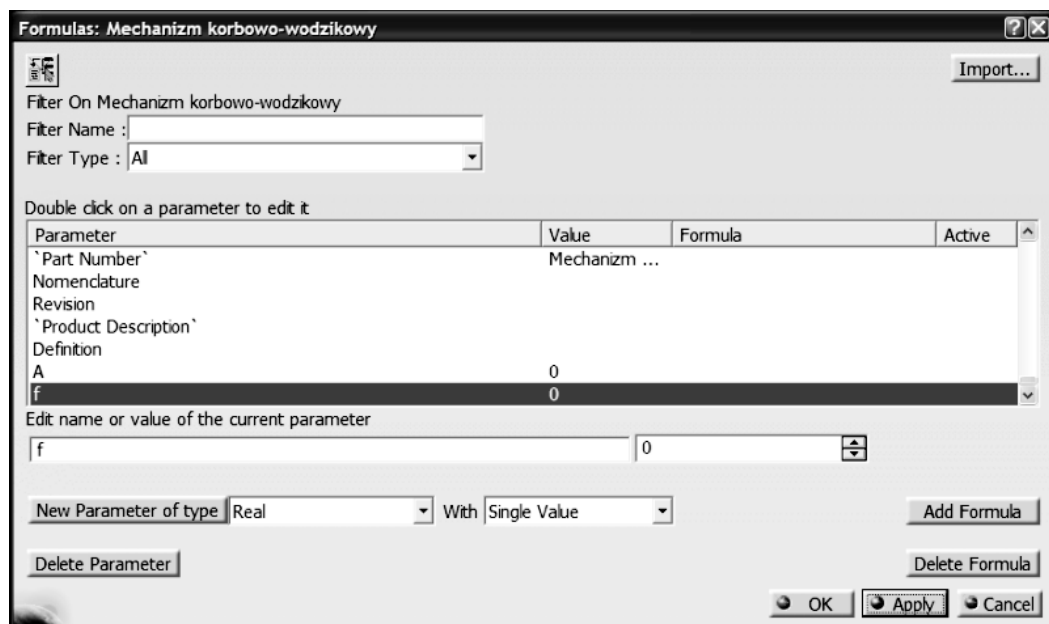
Jako drugą utwórzmy zmienną również typu *Real*, ale o nazwie *f* (rysunek 5.54). Będzie ona oznaczała **częstotliwość** ruchu harmonicznego.



Rysunek 5.52. Widok postaci modelu mechanizmu korbowo-wodzikowego



Rysunek 5.53. Utworzenie zmiennej typu Real o nazwie A

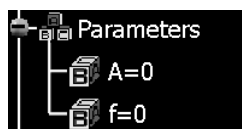


Rysunek 5.54. Utworzenie zmiennej typu *Real* o nazwie *f*

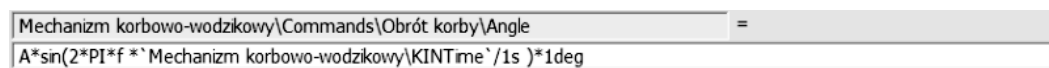
Utworzone parametry widzimy w strukturze drzewa topologicznego. Ich wartości domyślne są zerowe (rysunek 5.55).

Rysunek 5.55.

Widok parametrów *A*
oraz *f* w strukturze
drzewa topologicznego



Po utworzeniu niezbędnych parametrów zajmiemy się zdefiniowaniem formuły matematycznej. Treść formuły musi być zgodna ze wzorem opisującym wychylenie w ruchu harmonicznym, tj. $y(t) = A \times \sin(2 \times \pi \times f \times t + \varphi)$. Dla ułatwienia przyjmiemy, że kąt wychylenia początkowego jest zerowy ($\varphi = 0^\circ$). Postać tej formuły w zapisie charakterystycznym dla systemu *CATIA* widoczna jest na rysunku 5.56.



Rysunek 5.56. Widok postaci formuły opisującej ruch harmoniczny



Liczba π została predefiniowana w systemie *CATIA*, a jej użycie polega na zastosowaniu liter „PI”.

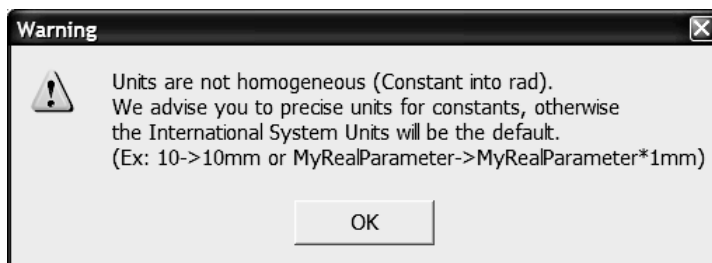


Funkcja $\sin()$ jest funkcją zaimplementowaną w systemie *CATIA*.

Po zapisaniu treści formuły i wybraniu przycisku *OK* (okno *Formulas Editor*) system informuje użytkownika o niejednorodności typów jednostek (rysunek 5.57). W naszym przypadku można nie przejmować się tym ostrzeżeniem, gdyż dzięki zastosowaniu końcowego mnożnika ($\times 1\text{deg}$) jednostką wyniku prawej strony formuły będzie stopień ($^{\circ}$).

Rysunek 5.57.

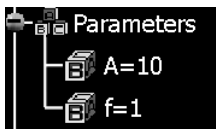
Widok zawartości
okna ostrzegawczego



Aby zastosowanie utworzonej formuły dało spodziewane efekty, musimy zmienić wartości utworzonych wcześniej zmiennych A i f , zgodnie z rysunkiem 5.58. Wartość częstotliwości wahań ustalamy na 1 (w domyśle 1 Hz), a więc w ciągu 1 s korba (oraz pozostałe ruchome człony mechanizmu) wykona jedno pełne wahnięcie. Wartość amplitudy ustalamy przykładowo na 10 (w domyśle 10 mm).

Rysunek 5.58.

Zaktualizowane
wartości
zmiennych A i f



Można zatem przystąpić do wykonania symulacji ruchu. Podczas symulacji warto dokonać obserwacji parametrów ruchu. W tym celu uaktywniamy okno *Sensors* (opcja *Activate sensors* okna *Kinematics Simulation*). Jako więc obserwowany zaznaczamy więz obroty *Revolute.1*, sterujący ruchem wahadłowym modelu korby (rysunek 5.59).

Potwierdzeniem prawidłowości zapisu formuły sterującej ruchem mechanizmu jest sinusoidalny wykres (rysunek 5.60). Widzimy, że w czasie 1 s powstała jedna pełna sinusoida, więc korba wykonała jedno pełne wahnięcie.

Posługując się utworzonymi parametrami, można poprzez zmianę ich wartości otrzymać inne realizacje ruchu wahadłowego. Dla przykładu: zwiększając wartość częstotliwości do 2 Hz , otrzymujemy sinusoidę, jak na rysunku 5.61.