

Mechanism Design and Analysis

Using PTC® Creo® Mechanism 6.0



Kuang-Hua Chang, Ph.D.

Visit the following websites to learn more about this book:

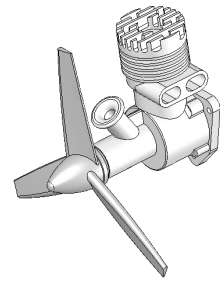


[amazon.com](https://www.amazon.com)

[Google books](https://books.google.com)

[BARNES & NOBLE](https://www.barnesandnoble.com)

Lesson 1: Introduction to *Mechanism*



1.1 Overview of the Lesson

The purpose of this lesson is to provide you with a brief overview on the *PTC Creo Mechanism* software tool or simply *Mechanism*. *Mechanism* is a virtual prototyping tool that supports mechanism design and analysis, which is also called motion analysis in this book. Instead of building and testing physical prototypes of the mechanism, you may use the *Mechanism* software tool to evaluate and refine the mechanism before finalizing the design and entering the functional prototyping stage. *Mechanism* will help you analyze and eventually design better engineering products. More specifically, the software enables you to size motors and actuators, determine power consumption, layout linkages, develop cams, understand gear trains, size springs and dampers, and determine interference between parts, which would usually require tests of physical prototypes. With such information, you will gain insight on how the mechanism works and why it behaves in certain ways. You will be able to modify the design and often achieve better design alternatives using the more convenient and less expensive virtual prototypes. In the long run, using virtual prototyping tools, such as *Mechanism*, will help you become a more experienced and competent design engineer.

In this lesson, we start with a brief introduction to *Mechanism* and the types of physical problems that *Mechanism* is capable of solving. We will then discuss capabilities offered by *Mechanism* for creating motion models, conducting motion analyses, and viewing motion analysis results. In the last section, we will mention examples employed in this book and major topics to be discussed in these examples.

Note that materials presented in this lesson will be brief. More details on various aspects of mechanism design and analysis using *Mechanism* will be given in later lessons.

1.2 What is *Mechanism*?

Mechanism is a computer software tool that enables engineers to analyze and design mechanisms. *Mechanism* is a module of the *PTC Creo* product family developed by *Parametric Technology Corporation*. This software allows users to create virtual mechanisms that answer general questions in product design such as those described next. An internal combustion engine shown in Figures 1-1 and 1-2 is used to illustrate some of the typical questions in mechanism design.

1. Will the components of the mechanism collide in operation? For example, will the connecting rod collide with the inner surface of the piston or the inner surface of the engine case during operation?
2. Will the components in the mechanism you design move according to your intent? For example, will the piston stay entirely in the piston sleeve? Will the system lock up when the firing force aligns vertically with the connecting rod?

3. How fast will the components move, e.g., the vertical motion of the piston?
4. How much torque or force does it take to drive the mechanism? For example, what will be the minimum firing load to move the piston? Note that in this case, proper friction forces must be added to simulate the resistance of the mechanism before a realistic firing force can be calculated.
5. What is the reaction force or torque generated at a connection (also called *joint* or *constraint*) between components (or bodies) during motion? For example, what is the reaction force at the joint between the connecting rod and the piston pin? This reaction force is critical since the structural integrity of the piston pin and the connecting rod must be ensured; i.e., they must be strong and durable enough to sustain the firing load in operation.

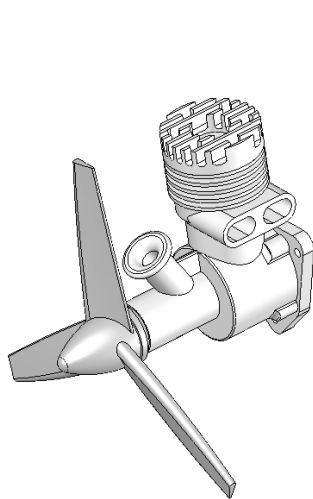


Figure 1-1 Internal Combustion Engine (Unexploded View)

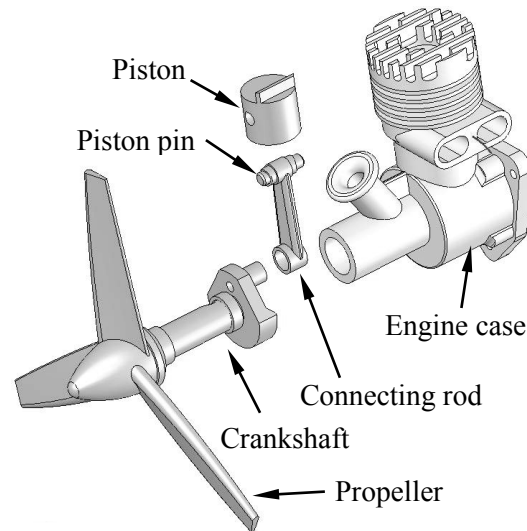


Figure 1-2 Internal Combustion Engine (Exploded View)

The modeling and analysis capabilities in *Mechanism* may help you answer these common questions accurately and realistically, as long as the motion model is properly defined. The capabilities available in *Mechanism* also help you search for better design alternatives. A better design alternative is very much problem dependent. It is critical that a design problem be clearly defined by the designer up front before searching for better design alternatives. For the engine example, a better design alternative can be a design that reveals:

1. A smaller reaction force applied to the connecting rod, and
2. No collisions or interference between components in motion.

In order to vary component sizes for exploring better design alternatives, the parts and assembly must be adequately parameterized to capture design intents. At the parts level, design parameterization implies creating solid features and relating dimensions properly. At the assembly level, design parameterization involves defining assembly constraints (called placement constraints in *Creo*) and relating dimensions within a single part or across parts. When a solid model is fully parameterized, a change in dimension value can be automatically propagated to all parts affected. Parts affected must be rebuilt successfully, and at the same time, they will have to maintain proper position and orientation with respect to one another without violating any assembly constraints or revealing part penetration or excessive gaps. For example, in this engine example, a change in the bore diameter of the engine case will alter not only the geometry of the case itself, but all other parts affected, such as the piston, piston sleeve,

and even the crankshaft, as illustrated in Figure 1-3. Moreover, they all have to be rebuilt properly and the entire assembly must stay intact through assembly constraints.

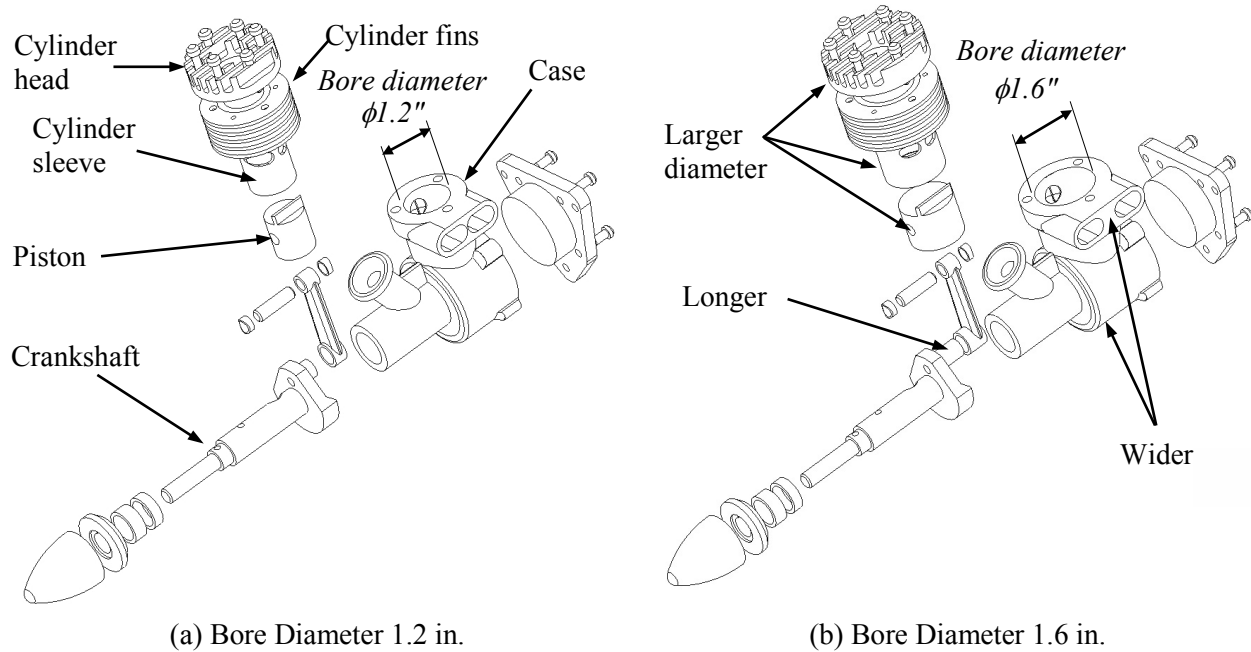
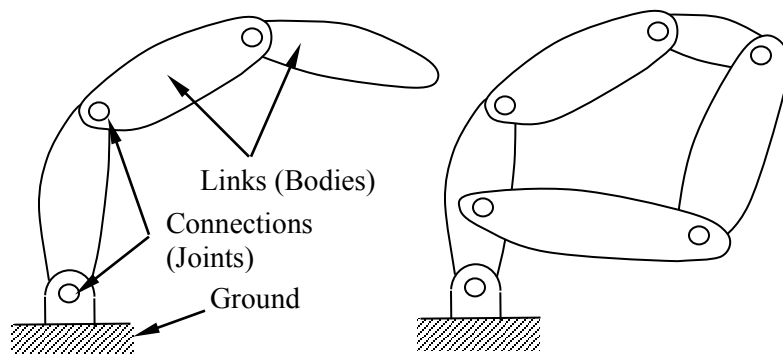


Figure 1-3 Parameterization of the Internal Combustion Engine—Exploded View

1.3 Mechanism and Motion Analysis

A mechanism is a mechanical device that transfers motion and/or force from a source to an output. It can be an abstraction (simplified model) of a mechanical system represented as a linkage. A linkage consists of links (or bodies), which are connected by connections (or joints), such as a pin joint, to form open or closed chains (or loops; see Figure 1-4). Such kinematic chains, with at least one link fixed, become mechanisms. In this book, all links are assumed rigid. In general, a mechanism can be represented by its corresponding schematic drawing. For example, a slider-crank mechanism represents the engine motion, as shown in Figure 1-5, which is a closed loop mechanism.



(a) Open Loop Mechanism (b) Closed Loop Mechanism

Figure 1-4 General Mechanisms

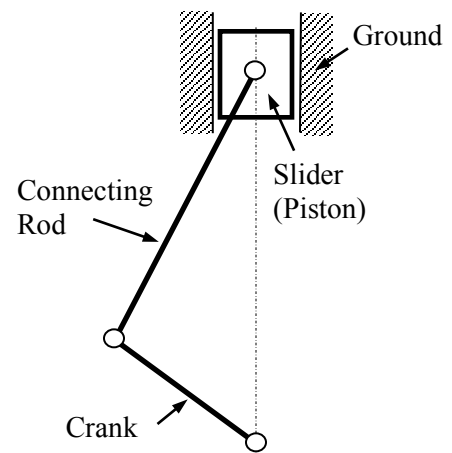


Figure 1-5 Schematic View of the Engine Motion Model

In general, there are two types of motion problems (among others) that you will need to solve in order to answer general questions regarding mechanism analysis and design. They are kinematic and dynamic problems.

Kinematics is the study of motion without regard for the forces that cause the motion. A kinematic mechanism must be driven by a servo motor (or driver) so that the position, velocity, and acceleration of each link in the mechanism can be analyzed at any given time. Typically, a kinematic analysis must be conducted before dynamic behavior of the mechanism can be simulated properly.

Dynamics is the study of motion in response to externally applied loads. The dynamic behavior of a mechanism is governed by Newton's laws of motion. The simplest dynamic problem is the particle dynamics covered in Sophomore Dynamics—for example, a spring-mass-damper system shown in Figure 1-6. In this case, motion of the mass is governed by the following equation derived from Newton's second law,

$$\Sigma F = p(t) - kx - c\dot{x} = m\ddot{x} \quad (1.1)$$

where (*) appearing on top of the physical quantity represents time derivative of the quantity, m is the total mass of the block, k is the spring constant, and c is the damping coefficient.

For a rigid body, mass properties (such as the total mass, center of mass, mass moment of inertia, etc.) are taken into account for dynamic analysis. For example, motion of a pendulum shown in Figure 1-7 is governed by the following equation of motion,

$$\Sigma M = -mg\ell \sin\theta = J\ddot{\theta} = m\ell^2\ddot{\theta} \quad (1.2)$$

where M is the external moment (or torque), J is the mass moment of inertia of the pendulum, m is the pendulum mass, g is the gravitational acceleration, and $\ddot{\theta}$ is the angular acceleration of the pendulum.

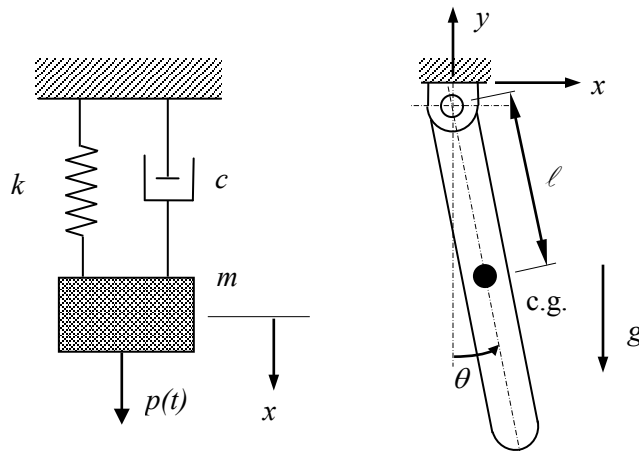


Figure 1-6 The Spring-Mass-Damper System

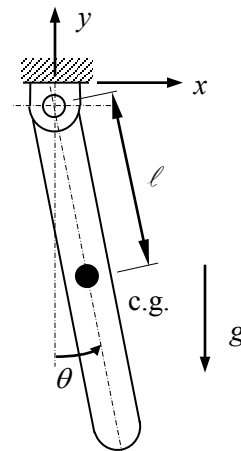


Figure 1-7 A Simple Pendulum

Dynamics of a multi-rigid body system, such as those illustrated in Figure 1-4, is a lot more complicated than the single body problems. Usually, a system of differential and algebraic equations governs the motion and the dynamic behavior of the system. Newton's law must be obeyed by individual bodies in the system at all times. The motion of the system will be determined by the loads acting on the bodies or joint axes (e.g., a torque driving the system). Reaction forces and/or torques at the joint connections hold the bodies together.

Note that in *Mechanism*, you may create a kinematic analysis model, for example, using a servo motor to drive the mechanism before carrying out a dynamic analysis. In this case, position, velocity, and acceleration results may be similar to those of dynamic analysis; however, the inertia of the bodies will be taken into account for dynamic analysis; therefore, reaction forces among other physical quantities will be calculated between bodies.

1.4 Mechanism Capabilities

Overall Process

The overall process of using *Mechanism* for designing or analyzing a mechanism consists of three main steps: model creation, analysis, and result visualization, as illustrated in Figure 1-8. Key entities that constitute a motion model include a ground body that is always fixed, bodies that are movable, connections (or joints) that connect bodies, initial conditions that define the initial configuration of the mechanism, servo motors (or drivers) that drive the mechanism for kinematic analysis, and forces and torque that move the components of the mechanism. Most importantly, assembly placement constraints (or assembly mates) must be properly defined for the mechanism so that the motion model captures essential characteristics and closely resembles the behavior of the physical mechanism. More details about these entities are discussed later in this lesson.

The analysis capabilities in *Mechanism* include position (initial assembly), static (equilibrium configuration), kinematic, dynamic, and force balance (to retain the system in a prescribed configuration). For example, a static analysis brings bodies to an equilibrium configuration of the mechanism. More about the analysis capabilities in *Mechanism* will be discussed later in this lesson.

The analysis results can be visualized in various forms. You may animate motion of the mechanism, or generate graphs for more specific information, such as the reaction force of a joint in time domain. You may also query results at specific locations for a prescribed time frame. Furthermore, you may ask for a report on results that you specified, such as the acceleration of a moving body in the time domain.

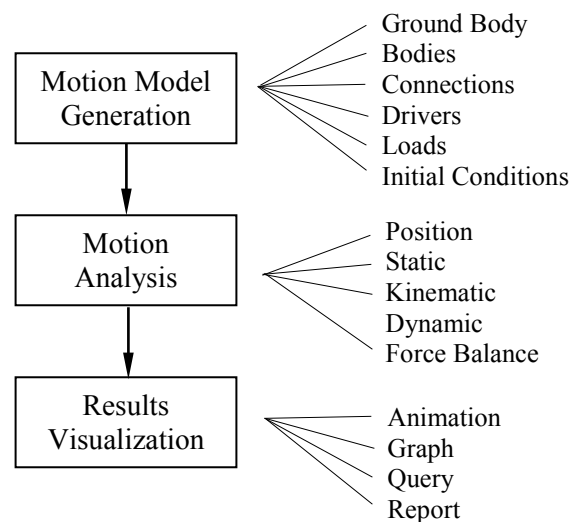


Figure 1-8 General Process of Using *Mechanism*

Operation Mode




Mechanism is embedded into *PTC Creo* (or *PTC Creo Parametric*). It is indeed an integrated module of *Creo*, and transition from *Creo* to *Mechanism* is seamless. All the solid models, assembly constraints, etc. defined in *Creo* are automatically carried over into *Mechanism*. *Mechanism* can be accessed through menus and windows inside *Creo*. The same assembly is used in both *Creo* and *Mechanism*.

Body geometry is essential for mass property computations in motion analysis. In *Mechanism*, all mass properties are ready for use. In addition, the detailed part geometry needed for interference checking is readily available.

User Interface

User interface of the *Mechanism* is identical to that of *Creo*, as shown in Figure 1-9. *PTC Creo* users should find it is straightforward to maneuver in *Mechanism*.

The *Creo* window consists of the following essential elements: command ribbon, *File* pull-down menu, toolbars, navigation area, *Graphics* window, shortcut menus, browser, and status bar.

The ribbon contains the command buttons organized within a set of tabs. On each tab, the related buttons are grouped. Each button on the ribbon consists of an icon and a label. For example, under the *Mechanism* tab, *Mechanism Analysis* , *Playback* , and *Measures*  buttons are grouped under *Analysis*. The major command buttons in *Mechanism* and their functions are summarized in Table 1-1.

The *File* (pull-down) menu located at the upper-left corner of the *Creo* window supports functions, such as *Open*, *Save*, and *Close*, for managing files and models, and preparing models for distribution.

The *Quick Access* toolbar is available regardless of which tab is selected on the ribbon. By default, it is located at the top of the *Creo* window. It provides quick access to frequently used buttons, such as buttons for opening and saving files, undo, redo, regenerate, close windows, switch windows, and so on. In addition, you can customize the *Quick Access* toolbar to include other frequently used buttons and cascading lists from the ribbon.

The *Graphics* toolbar is embedded at the top of the *Graphics* window. The buttons on the *Graphics* toolbar control the display of model. You can hide or display the buttons on the toolbar. You can change the location of the toolbar by right-clicking and choosing a location from the shortcut menu.

The shortcut menu is contextual user interface relative to the selected object. Press the right mouse button until you see the shortcut menu, for example, the servo motor shown in Figure 1-9.

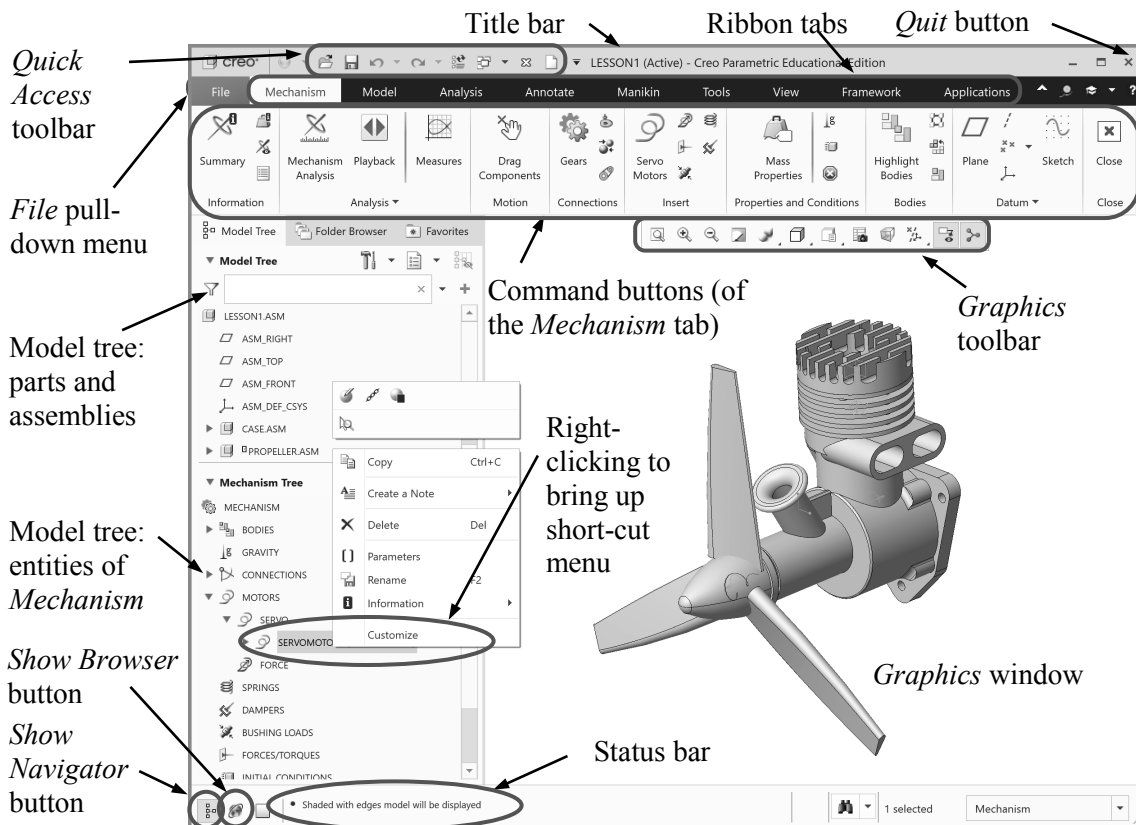




Figure 1-9 User Interface of *Mechanism*

The navigator on the left of the *Graphics* window includes the *Model Tree*, *Layer Tree*, *Detail Tree*, *Folder* browser, and *Favorites*. The *Model Tree* tab  on the status bar (located at the lower left corner of the *Creo* window) controls the display of the navigator. When the *Mechanism* button  is clicked under the *Applications* tab, the model tree is split into two; entities of parts and assemblies are listed in the upper tree, and entities of *Mechanism* are listed in the lower tree, as shown in Figure 1-9.

The *Graphics* window, to the right of the navigator, displays the motion model with which you are working.





















The *Creo* browser provides access to internal and external Web sites. Clicking the *Show Browser* button  next to the status bar controls the display of the browser.

Table 1-1 The Major Buttons and Groups of *Mechanism*

Group	Symbol	Name	Function
Analysis		Mechanism Analysis	Define and run an analysis.
		Playback	Play back the results of your analysis run. You can also save or export the results or restore previously saved results.
		Measures	Create measures, and select measures and result sets to display. You can also graph the results or save them to a table.
Motion		Drag Components	Drag components in an assembly.
Connections		Gears	Create gear pairs.
		Cams	Create a cam-follower connection.
		3D Contacts	Define a 3D contact between two parts in different bodies.
		Belts	Create a belt and pulley system.
Inserts		Servo Motors	Define a servo motor (driver).
		Force Motors	Define a new force motor.
		Forces/Torques	Define a force or a torque.
		Bushing Loads	Create a Bushing Load feature to simulate movement between two bodies.
		Springs	Define a new spring.
		Dampers	Define a new damper.
Properties and Conditions		Mass Properties	Specify mass properties for a part or specify density for an assembly.
		Gravity	Define gravity.
		Initial Conditions	Specify initial position snapshots, and define the velocity initial conditions for a point, motion axis or body.
		Termination Conditions	Specify conditions to terminate the motion analysis.

The status bar is located at the bottom of the *Creo* window. When applicable, the status bar displays the controls, such as the display of the navigation area (by clicking the *Show Navigator* button ) and one-line messages related to work in a window. You may right-click in the message area and then click *Message Log* to review past messages.

Defining Motion (or Motion Simulation) Entities

The basic entities of a motion (also called simulation in this book) model created in *Mechanism* consist of ground, bodies, connections, initial conditions, drivers, and loads. Each of the basic entities will be briefly introduced. More details can be found in later lessons.

Ground Body

A ground (or ground body) represents a fixed entity in space. The root assembly is always fixed; therefore, it becomes ground body by default (or part of the ground body). Also, the datum coordinate system of the root assembly is assigned as the *WCS* (World Coordinate System). All datum features and parts fixed to the root assembly are part of the ground body.

Bodies

A body represents a single rigid component (or link) that moves relative to the other body (or bodies in some cases). A body may consist of several *Creo* parts fully constrained using placement constraints. A body contains a local coordinate system (*LCS*), body points (created as datum points), and mass properties. Note that body points are created for defining connections, force applications, etc.

A spatial body consists of three translational and three rotational degrees of freedom (dofs). That is, a rigid body can translate and rotate along the *X*-, *Y*-, and *Z*-axes of a coordinate system. Rotation of a rigid body is measured by referring the orientation of its *LCS* to *WCS*, which is fixed to the ground body.

In *Mechanism*, the *LCS* is assigned automatically, usually, to the default datum coordinate system of the body (either part or assembly), and the mass properties are calculated using part geometry and material properties referring to the *LCS*. Datum axes and points are essential in creating a motion model since they are employed for defining connections and the location of external load application.

Connections

A connection in *Mechanism* can be a joint, cam, or gear that connects two bodies. Typical joints include pin, slider, bearing, ball, cylinder, etc. A connection constrains the relative motion between bodies. Each independent movement permitted by a connection is called a (free) degree of freedom (dof). The degrees of freedom that a connection allows can be translation and rotation along three perpendicular axes, as shown in Figure 1-10.

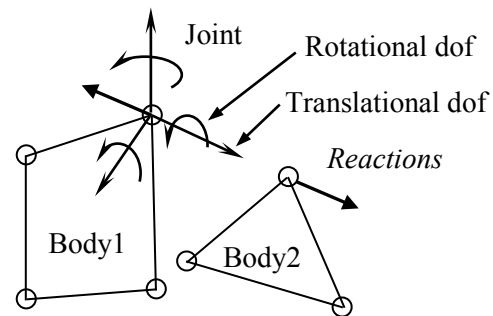


Figure 1-10 A Typical Joint in *Mechanism*

There are two types of options available for assembling parts: *User Defined* (joint) and *Automatic* (placement). The *User Defined* type consists of commonly employed joints for defining motion models, such as rigid, pin, ball, bearing, planar, cylinder joints, etc., which directly correspond to connections implemented in the physical mechanism. The *Automatic* constraints are placement constraints we employed for assembling parts, such as coincident, concentric, etc. Both types eliminate prescribed

degrees of freedom between components. For example, a ball joint may be created simply by coinciding two datum points in their respective bodies, allowing all three rotational dofs. Some of the joints are directly equivalent to placement constraint. For example, mating (or aligning, coinciding) two planar faces is equivalent to defining a planar joint. However, instead of completely fixing all the movements, certain dofs (translations and/or rotations) are left to allow the desired movement for a motion model.

The connections produce equal and opposite reactions (forces and/or torques) on the bodies connected. The symbol of a given joint tells the translational and/or rotational dof that the joint allows in regard to movement. Understanding the four basic joint symbols shown in Figure 1-11 will enable you to read existing joints in motion models. More details about joint types available in *Mechanism* will be discussed in later lessons. A complete list of joints available in *Mechanism* can be found in Appendix A.

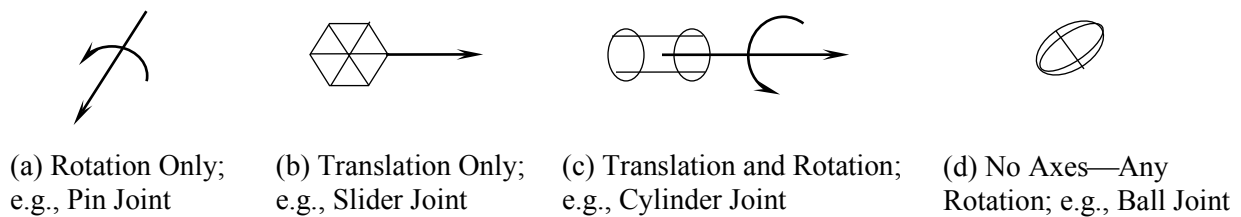


Figure 1-11 The Four Basic Joint Symbols

Degrees of Freedom

As mentioned earlier, an unconstrained body in space has six degrees of freedom: three translational and three rotational. When joints are added to connect bodies, constraints are imposed to restrict the relative motion between them. For example, a pin joint allows one rotational motion between bodies. As defined in the engine example shown in Figure 1-12, joint *Pin1* restricts movement on five dofs so that only one rotational motion is allowed between the propeller assembly and the ground body (*case.asm*).

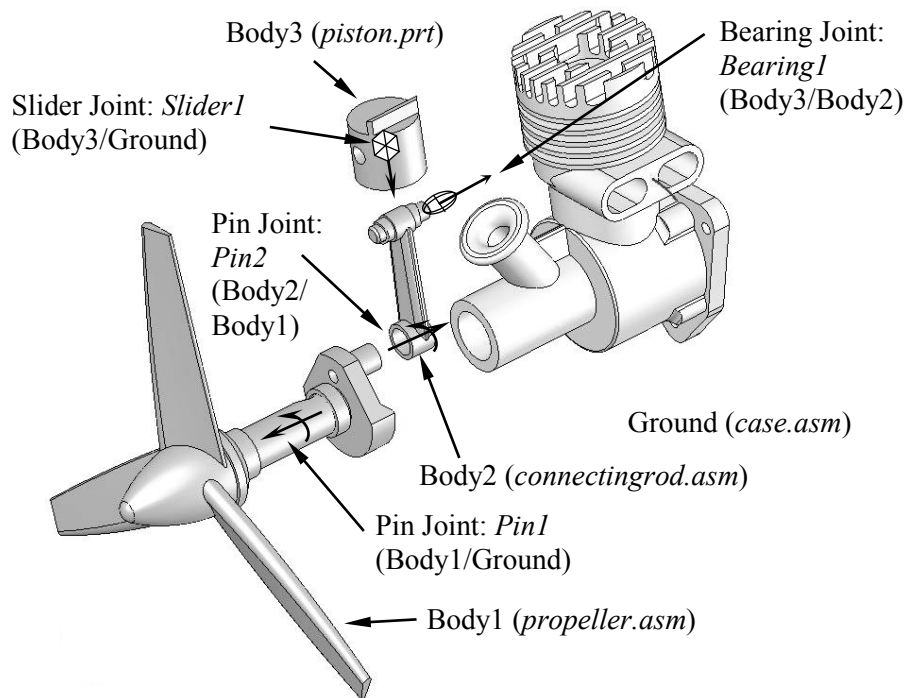


Figure 1-12 An Example Motion Model in Exploded View

For a given motion model, you can determine its number of degrees of freedom using the Gruebler's count. *Mechanism* uses the following equation to calculate the Gruebler's count:

$$D = 6M - N - O \quad (1.3)$$

where D is the Gruebler's count representing the overall free degrees of freedom of the mechanism, M is the number of bodies excluding the ground body, N is the number of dofs restricted by all joints, and O is the number of motion drivers (or servo motors) defined in the system.

The single-piston engine shown in Figure 1-12 consists of three bodies (excluding the ground body), two pin joints, one slider joint, and one bearing joint. A pin or slider joint removes five degrees of freedom, and a bearing joint removes two dofs. In addition, a servo motor is added to the rotational dof of the joint *Pin1*. Therefore, according to Eq. 1.3, the Gruebler's count for the engine example is

$$D = 6 \times 3 - (3 \times 5 + 1 \times 2) - 1 \times 1 = 0$$

In general, a valid motion model should have a Gruebler's count of 0. However, in creating motion models, some joints remove redundant dofs. For example, two hinges, modeled using two pin joints, support a door. The second pin joint adds five redundant dofs. The Gruebler's count becomes

$$D = 6 \times 1 - 2 \times 5 = -4$$

For kinematic analysis, the Gruebler's count must be equal to or less than 0. The solver recognizes and deactivates redundant constraints during analysis. For a kinematic analysis, if you create a model and try to animate it with a Gruebler's count greater than 0, the animation will not run and an error message will appear.

If the Gruebler's count is less than zero, the solver will automatically remove redundancies. In this engine example, if the bearing joint between the connecting rod and the piston pin is replaced by a pin joint, the Gruebler's count becomes

$$D = 6 \times 3 - 4 \times 5 - 1 \times 1 = -3$$

To get the Gruebler's count to zero, it is often possible to replace joints that remove a large number of constraints with joints that remove a smaller number of constraints and still restrict the mechanism motion in the same way. *Mechanism* detects the redundancies and ignores redundant dofs in all analyses. In dynamic analysis, the redundancies lead to an outcome with a possibility of incorrect reaction results, yet the motion is correct. For complete and accurate reaction forces, it is critical that you eliminate redundancies from your mechanism. The challenge is to find the joints that will impose non-redundant constraints and still allow for the intended motion. When this is not feasible, cautions must be exercised in checking results of dynamic simulation, especially for reaction forces. Examples included in this book should give you some ideas in choosing proper joints.

Loads

Loads are used to drive a mechanism. Physically, loads are produced by motors, springs, dampers, gravity, etc. A load entity in *Mechanism* can be a force or torque. The force and torque are represented by an arrow and double-arrow symbols, respectively, as shown in Figures 1-13 and 1-14. Note that a load can be applied to a body, a point in a body, or between two points in different bodies.

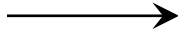


Figure 1-13 The Force Symbol

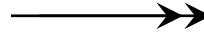


Figure 1-14 The Torque Symbol

Drivers or Servo Motors

Drivers or servo motors are used to impose an intended motion on a mechanism. Servo motors cause a specific type of motion to occur between two bodies in a single degree of freedom. Servo motors specify position, velocity, or acceleration as function of time; and can control either translational or rotational motion. The driver symbol is shown in Figure 1-15.

Note that a driver must be defined along a movable axis of the joint you select. Otherwise, no motion will occur. When properly defined, drivers will account for the remaining dofs of the mechanism calculated using Eq. 1.3.

An example of a motion model created using *Mechanism* is shown in Figure 1-12. In this engine example, twenty-six *Creo* parts are grouped into four bodies. In addition, four joints plus a driver are defined for a kinematic analysis. You may open this model and animate its motion by following the steps described in Section 1.5.

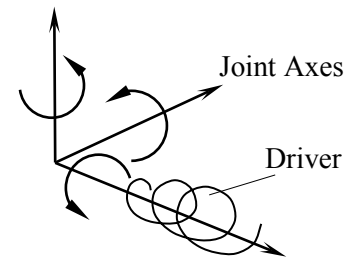


Figure 1-15 The Driver (Servo motor) Symbol

Types of Mechanism Analyses

There are five analysis options supported in *Mechanism*: position, static, kinematic, dynamic, and force balance.

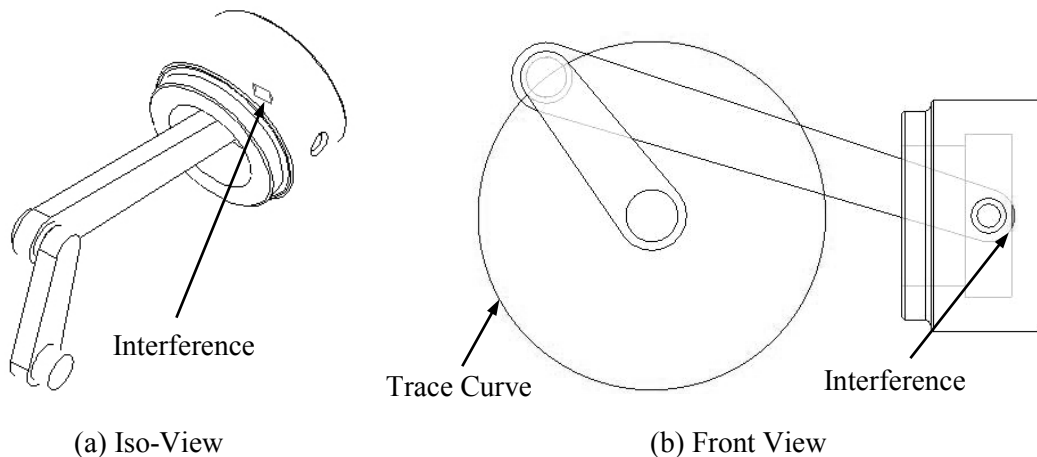


Figure 1-16 Trace Curve and Interference Checking of a Position (or Assembly) Analysis for a Slider-Crank Mechanism

A position analysis is a series of assembly analyses driven by servo motors. Only motion axes or geometric servo motors can be included in position analyses. Force motors do not appear in the list of possible motor selections when adding a motor for a position analysis. A position analysis simulates the mechanism's motion, satisfying the requirements of your servo motor profiles and any joint, cam-follower, slot-follower, or gear-pair connections, and records position data for the mechanism's various

components. It does not take force and mass into account when performing the analysis. Therefore, you do not have to specify mass properties for your mechanism. Dynamic entities in the model, such as springs, dampers, gravity, forces/torques, and force motors, do not affect a position analysis. You may use a position analysis to study positions of components over time, interference between components, and trace curves of the mechanism's motion (for example, see Figure 1-16).

Static analysis is used to find the rest position (equilibrium condition) of a mechanism, in which none of the bodies are moving. Static analysis is related to mechanical advantage—for example, how much load can be resisted by a driving motor. A simple example of the static analysis is shown in Figure 1-17.

Motion analysis consists of mainly kinematic and dynamic analyses. As discussed earlier, kinematics is the study of motion without regard for the forces that cause the motion. A mechanism can be driven by a servo motor for a kinematic analysis, where the position, velocity, and acceleration of each link of the mechanism can be analyzed at any given time. For example, a servo motor drives a mechanism shown in Figure 1-18 at a constant angular velocity ω .

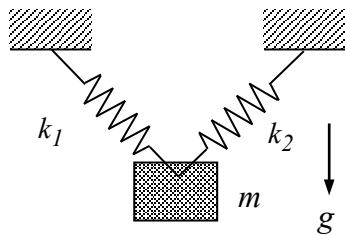


Figure 1-17 Static Analysis

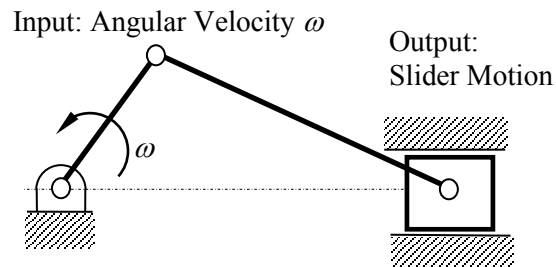


Figure 1-18 Kinematic Analysis

Dynamic analysis is used to study the mechanism motion in response to loads, as illustrated in Figure 1-19. This is the most complicated and common but usually more time-consuming analysis.

Force balance calculates the required force and torque to retain the system in a prescribed configuration.

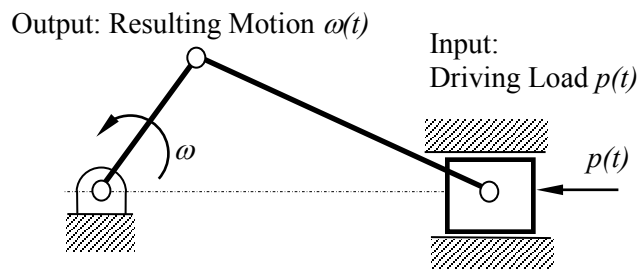


Figure 1-19 Dynamic Analysis

Viewing Results

In *Mechanism*, results of the motion analysis can be realized using animations, graphs, reports, and queries. Animations show the configuration of the mechanism in consecutive time frames. Animations will give you a global view on how the mechanism behaves; for example, a motion animation of a single-piston engine shown in Figure 1-20.

You may choose a joint or a datum point in the motion model to generate result graphs; for example, the graph in Figure 1-21 shows the angular position of a simple pendulum example (please see *Lesson 4* for more details). These graphs give you a quantitative understanding of the behavior of the mechanism. You may also pick a data point on the graph to query the results of your interest at a specific time frame. In addition, you may ask *Mechanism* for a report that includes a complete set of results output in the form of numerical data.

In addition to the capabilities discussed above, *Mechanism* allows you to check interference between bodies during motion (more to be discussed in *Lesson 5*). Furthermore, the reaction forces calculated can be used to support structural analysis using, for example, *Creo Simulate*, a p-version finite element analysis module of *PTC Creo*.

1.5 Open Lesson 1 Model

A motion model for the single piston engine model shown in Figure 1-1 has been created for you. Download the files from www.sdcpublications.com, unzip them, and locate the engine assembly file (*lesson1.asm*) under *Lesson 1* folder. Copy *Lesson 1* folder to your computer.

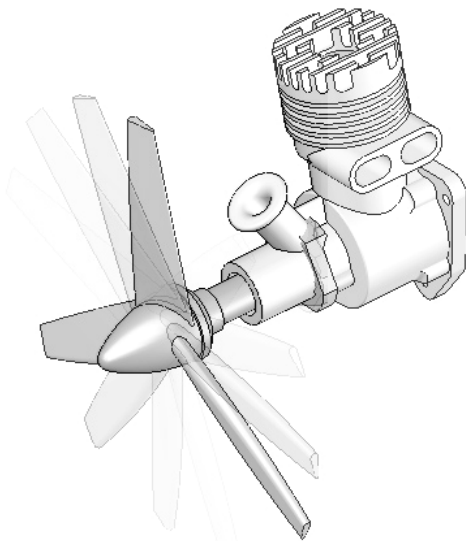


Figure 1-20 Motion Animation

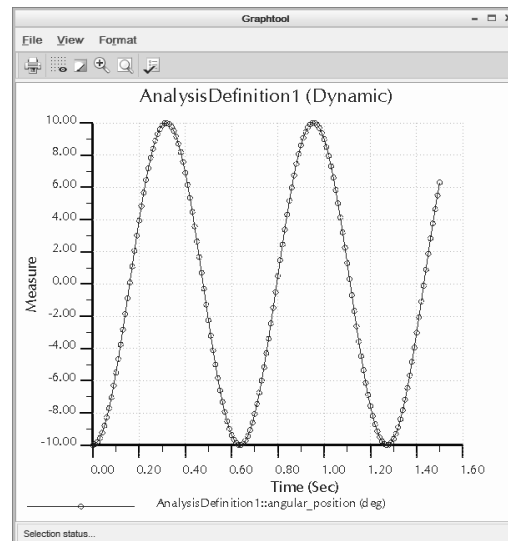








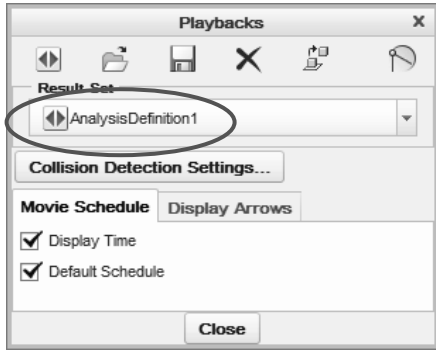
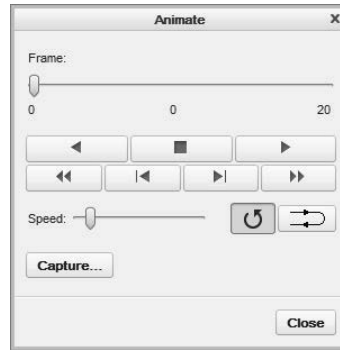
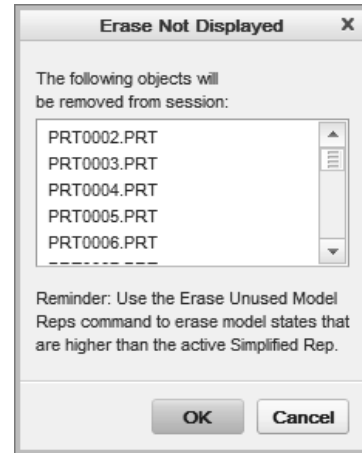
Figure 1-21 A Sample Result Graph

Start *PTC Creo*, select *Lesson 1* folder as the working directory by choosing *File > Manage Session > Select Working Directory*, and open the assembly model: *lesson1.asm*. You should see an assembled engine model similar to that of Figure 1-1.

To enter *Mechanism*, simply click the *Applications* tab on top of the *Graphics* window, and click the *Mechanism* button .

You should see *Mechanism* window layout similar to that of Figure 1-9. To replay results click the *Playback* button  on top. The *Playbacks* dialog box (Figure 1-22) appears. In the *Playbacks* dialog box, click the *Open* button , and select the previously saved playback file *AnalysisDefinition1.pbk* (this file is included in the *Lesson 1* folder). Click the *Play current result set* button  at the top left corner. The *Animate* dialog box (Figure 1-23) opens. Click the *Play* button  to play the motion of the engine. You should see the motion animation similar to that of Figure 1-20.

To close the model, choose *File > Close*. From the *Home* tab, click the *Erase Not Displayed* button  in the *Data* group. The *Erase Not Displayed* dialog box opens (Figure 1-24). Click *OK* to erase all parts and assemblies temporarily stored in the memory.

Figure 1-22 The *Playbacks* Dialog BoxFigure 1-23 The *Animate* Dialog BoxFigure 1-24 The *Erase Not Displayed* Dialog Box

1.6 Motion Examples

Numerous motion examples will be introduced in this book to illustrate step-by-step details of modeling, analysis, and result visualization capabilities in *Mechanism*. We will start with a simple ball throwing example. This example will give you a quick start and a quick run-through on *Mechanism*. *Lessons 3 through 8* focus on modeling and analysis of basic mechanisms. In these lessons, you will learn various joint types, including pin, slider, rigid, etc.; connections, including springs, gears, cam-followers; drivers and forces; various analysis types; and measures and results. *Lessons 9 and 10* are application lessons, in which real-world mechanisms will be introduced to show you how to apply what you learn to more complicated applications. All examples and main topics to be discussed in each lesson are summarized in the Table 1-2.

Table 1-2 Summary of Lessons and Motion Examples in this Book

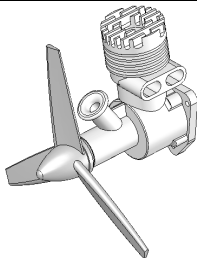
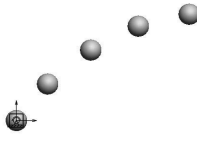
Lesson	Title	Example	Problem Type	Things to Learn
1	Single-Piston Engine		Multibody Kinematic Analysis	1. General introduction to <i>Creo Mechanism</i>
2	Ball Throwing Example		Particle Dynamics	<ol style="list-style-type: none"> 1. This lesson offers a quick run-through on modeling and analysis capabilities in <i>Mechanism</i>. 2. You will learn the process of using <i>Mechanism</i> to construct a motion model, run analysis, and visualize the motion analysis results. 3. Simulation results are verified using analytical equations of motion.

Table 1-2 Summary of Lessons and Motion Examples in this Book (Cont'd)

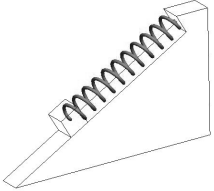







Lesson	Title	Example	Problem Type	Things to Learn
3	Spring-Mass System		Particle Dynamics	<ol style="list-style-type: none"> 1. This is a classical spring-mass system example you learned in <i>Dynamics</i>. 2. You will learn how to create a mechanical spring, align the block with the slope surface, and add an external force to pull the block. 3. Simulation results are verified using analytical equations of motion.
4	A Simple Pendulum		Particle Dynamics	<ol style="list-style-type: none"> 1. This lesson goes more in-depth about creating joints in <i>Mechanism</i>. Pin and rigid joints will be introduced. 2. Simulation results are verified using analytical equations of motion.
5	A Slider Crank Mechanism		Multibody Kinematic and Dynamic Analyses	<ol style="list-style-type: none"> 1. This lesson uses a slider-crank mechanism to introduce more joint types as well as conduct position, kinematic, and dynamic analyses. 2. In addition to joints, you will learn to create drivers for a kinematic analysis. 3. The interference checking capability will be discussed. 4. Kinematic analysis results are verified using analytical equations of motion.
6	A Compound Spur Gear Train		Gear Train Analysis	<ol style="list-style-type: none"> 1. This lesson focuses on simulating motion of a spur gear train system. 2. You will learn how to use <i>Mechanism</i> to create gear connections, analyze the gear train, and define measures for gears. 3. Simulation results are verified using analytical equations.
7	Planetary Gear Train Systems		Planetary Gear Train Analysis	<ol style="list-style-type: none"> 1. This lesson is similar to <i>Lesson 6</i> but focuses on planetary gear trains. 2. Both single- and multi-gear systems will be discussed. 3. Some simulation results are found incorrect compared with those obtained from analytical equations.
8	Cam and Follower		Multibody Dynamic Analysis	<ol style="list-style-type: none"> 1. This lesson discusses cam and follower joint. 2. An inlet or outlet valve system of an internal combustion engine will be created and simulated. 3. Kinematic and dynamic characteristics of the valve will be monitored in the motion simulation of the system. 4. Design of the system will be discussed.

Table 1-2 Summary of Lessons and Motion Examples in this Book (Cont'd)

Lesson	Title	Example	Problem Type	Things to Learn
9	Assistive Device for Wheelchair Soccer Game		Multibody Dynamic Analysis	<ol style="list-style-type: none"> 1. This is an application lesson. This lesson shows you how to assemble and simulate motion of an assistive device for playing wheelchair soccer game. 2. Numerous joints, spring, and force will be created for the system. 3. Measures will be defined to assess the design of the system.
10	Kinematic Analysis for Racecar Suspension		Multibody Kinematic Analysis	<ol style="list-style-type: none"> 1. This is the second and the last application lesson of the book. A quarter of a racecar suspension will be employed for kinematic analyses. 2. A road profile will be modeled by using a cam with prescribed profile. The cam will be connected to the tire using a cam-follower connection. 3. Various measures, including the camber angle, will be introduced to assess the design of the suspension system.

Note that example files have been prepared for you to go through all the lessons. In addition to *PTC Creo* parts and assemblies, each lesson folder contains complete motion models as well as simulation result files. You may want to open the motion models and review the simulation results; e.g., play motion animations, to become more familiar with the simulations before going through the lessons.