Creating and Running a Dynamic Analysis

A force simulating the engine firing load (acting along the negative X-direction) will be added to the piston for a dynamic simulation. It will be more realistic if the force can be applied when the piston starts moving to the left (negative X-direction) and can be applied only for a selected short period. In order to do so, we will have to define measures that monitor the position of the piston for the firing load to be activated. Unfortunately, such a capability is not available in COSMOSMotion. Therefore, the force is simplified as a step function of 3 lb along the negative X-direction applied for 0.1 seconds. The force will be defined as a point force at the center point of the end face of the piston, as shown in Figure 5-21.

Before we add the force, we will turn off the angular velocity driver defined at the joint Revolute2 in the previous simulation. We will have to delete the simulation before we can make any changes to the simulation model. Delete the simulation result by clicking the Delete Results button at the bottom of the browser.

From the browser, expand the Constraints branch, and then the Joints branch. Right click Revolute to bring up the Edit Mate-Defined Joint dialog box (Figure 5-22). Pull-down the Motion Type and choose Free. Click Apply to accept the change.

Note that if you run a simulation now, nothing will happen since there is no motion driver or force defined (gravity has been turned off).

Now we are ready to add the force. The force can be added from the browser by expanding the Forces branch, right clicking the Action Only node, and choosing Add Action-Only Force, as shown in Figure 5-23. In the Insert Action-Only Force dialog box, the Select Component to which Force is Applied field (see Figure 5-24) will be active (highlighted in red) and ready for you to pick the component.

Pick the end face of the piston, as shown in Figure 5-21. The part piston-1 is now listed in the Select Component to which Force is Applied field, and piston-1/DDMFace10 is listed in both the Select Location and the Select Direction fields. That is, the force will be applied to the center of the end face and in the direction that is normal to the face; i.e., in the positive X-direction.

Figure 5-22 Turning Off the Motion Driver

Figure 5-23 The Insert Action-Only Force Dialog Box
Now in the Insert Action-Only Force dialog box (Figure 5-24), the Select Reference Component to orient Force field is active (highlighted in red) and is ready for selection. We will pick the bearing (ground) for reference. Or you may simply click the ground button \( \downarrow \) to the right of the field. After that, you should see *Assemi* appear in the Select Reference Component to orient Force field.

Click the Function tab (see Figure 5-25), choose Step for function, and enter the followings:

- **Initial Value**: -3
- **Final Value**: 0
- **Start Step Time**: 0
- **End Step Time**: 0.1

Note that the negative sign for the initial value is to reverse the force direction to the negative X-direction.

Click the graph button (right most, as shown in Figure 5-25), the step function will appear like the one in Figure 5-26. Note that this is a smoothed step function with time extrapolated to the negative domain. The force varies from \(-3\) at 0 second to 0 at 0.1 seconds. During the simulation, the force will be activated at the beginning; i.e., 0 second.

Close the graph and click Apply button to accept the force definition. You should see a force symbol added to the piston, as shown in Figure 5-4.

Before running a simulation, we will increase the number of frames in order to see more refined results and graphs. From the browser, right click the Motion Model node, and choose Simulation Parameters. Change the number of frames to 500. Accept the change and then right click the Motion Model node, and choose Run Simulation. The mechanism will move and the crank will make several turns before reaching the end of the simulation duration; i.e., 1 second by default.

Graphs created in previous simulation, such as the piston position, etc., should appear immediately at the end of the simulation. As shown in Figure 5-27, the piston moves along the negative X-direction for about 0.15 seconds before reversing its direction. It is also evident in the velocity graph shown in Figure 5-28 that the velocity changes signs at two instances (close to 0.15 and 0.48 seconds). Recall that the force was applied for the first 0.1 seconds. Had the force application lasted longer, the piston could be continuously pushed to the left (negative X-direction) even when the piston reaches the left end and tries to move to the right (due to inertia). As a result, the crank would have been oscillating at the left of the center of the bearing; i.e., between 0 and 180 degrees about the Z-axis, without making a complete turn.
Graph the reaction force for the applied force at the piston by expanding the Forces branch, then the Action Only branch, right clicking the ForceAO, and choosing Plot. The reaction force that represents the actual force applied to the piston appears, as shown in Figure 5-29. The graph shows that the force of 3 lbf was applied at the beginning of the simulation. The force gradually decreases to 0 in the 0.7-second period, which is what we expected and is consistent with the force function, as seen in Figure 5-26.

Graph the reaction force at the joint Revolute (between crank and the ground) along the X-direction; you should see a graph like that of Figure 5-30. There are three peaks in Figure 5-30 representing when the largest reaction forces occur at the joint, which occurs at close to 0.15, 0.48, and 0.8 seconds; i.e., when the piston reverses its moving direction. The results make sense.

Save your model. We will carry out theoretical calculations to verify the simulation results. We will focus on kinematic analysis.
5.4 Result Verifications

In this section, we will verify the motion analysis results using kinematic analysis theory often found in mechanism design textbooks. Note that in kinematic analysis, position, velocity, and acceleration of given points or axes in the mechanism, are analyzed.

In kinematic analysis, forces and torques are not involved. All bodies (or links) are assumed massless. Hence, mass properties defined for bodies are not influencing the analysis results.

The slider-crank mechanism is a planar kinematic analysis problem. A vector plot that represents the positions of joints of the planar mechanism is shown in Figure 5-31. The vector plot serves as the first step in computing position, velocity, and accelerations of the mechanism.

The position equations of the system can be described by the following vector summation,

\[ Z_1 + Z_2 = Z_3 \]  

(5.1)

where

\[ Z_1 = Z_1 \cos \theta_A + iZ_1 \sin \theta_A = Z_1 e^{i\theta_A} \]
\[ Z_2 = Z_2 \cos \theta_B + iZ_2 \sin \theta_B = Z_2 e^{i\theta_B} \]
\[ Z_3 = Z_3, \text{ since } \theta_C \text{ is always 0.} \]

The real and imaginary parts of Eq. 5.1, corresponding to the \( X \) and \( Y \) components of the vectors, can be written as

\[ Z_1 \cos \theta_A + Z_2 \cos \theta_B = Z_3 \]  

(5.2a)

\[ Z_1 \sin \theta_A + Z_2 \sin \theta_B = 0 \]  

(5.2b)

In Eqs. 5.2a and 5.2b, \( Z_1, Z_2, \) and \( \theta_1 \) are given. We are solving for \( Z_3 \) and \( \theta_B \). Equations 5.2a and 5.2b are non-linear. Solving them directly for \( Z_3 \) and \( \theta_B \) is not straightforward. Instead, we will calculate \( Z_3 \) first, using trigonometric relations; i.e.,

\[ Z_3^2 = Z_1^2 + Z_3^2 - 2Z_1Z_3 \cos \theta_A \]

Hence,

\[ Z_3^2 - 2Z_1 \cos \theta_A Z_3 + Z_1^2 - Z_3^2 = 0 \]

Solving \( Z_3 \) from the above quadratic equation, we have

\[ Z_3 = \frac{2Z_1 \cos \theta_A \pm \sqrt{(2Z_1 \cos \theta_A)^2 - 4(Z_1^2 - Z_3^2)}}{2} \]  

(5.3)
where two solutions of $Z_2$ represent the two possible configurations of the mechanism shown in Figure 5-32. Note that point C can be either at $C$ or $C'$ for any given $Z_1$ and $\theta_A$.

From Eq. 5.2b, $\theta_B$ can be solved by

$$\theta_B = \sin^{-1}\left(-\frac{Z_1 \sin \theta_A}{Z_2}\right)$$

(5.4)

Similarly, $\theta_A$ has two possible solutions, corresponding to vector $Z_j$.

Taking derivatives of Eqs. 5.2a and 5.2b with respect to time, we have

$$-Z_1 \sin \theta_A \dot{\theta}_A - Z_2 \sin \theta_B \dot{\theta}_B = \dot{Z}_3$$

(5.5a)

$$Z_1 \cos \theta_A \dot{\theta}_A + Z_2 \cos \theta_B \dot{\theta}_B = 0$$

(5.5b)

where $\dot{\theta}_A = \frac{d\theta_A}{dt} = \omega_A$ is the angular velocity of the rotation driver, which is a constant. Note that Eqs. 5.5a and 5.5b are linear functions of $\dot{Z}_3$ and $\dot{\theta}_B$. Rewrite the equations in a matrix form

$$\begin{bmatrix} Z_2 \sin \theta_B \\
Z_2 \cos \theta_B \end{bmatrix} \begin{bmatrix} \dot{\theta}_B \\
\dot{Z}_3 \end{bmatrix} = \begin{bmatrix} -Z_1 \sin \theta_A \dot{\theta}_A \\
-Z_1 \cos \theta_A \dot{\theta}_A \end{bmatrix}$$

(5.6)

Equation 5.6 can be solved by

$$\begin{bmatrix} \dot{\theta}_B \\
\dot{Z}_3 \end{bmatrix} = \begin{bmatrix} Z_2 \sin \theta_B & 1 \\
Z_2 \cos \theta_B & 0 \end{bmatrix}^{-1} \begin{bmatrix} -Z_1 \sin \theta_A \dot{\theta}_A \\
-Z_1 \cos \theta_A \dot{\theta}_A \end{bmatrix}$$

$$= \frac{1}{-Z_2 \cos \theta_B} \begin{bmatrix} 0 & -1 \\
-Z_2 \cos \theta_B & Z_2 \sin \theta_B \end{bmatrix} \begin{bmatrix} -Z_1 \sin \theta_A \dot{\theta}_A \\
-Z_1 \cos \theta_A \dot{\theta}_A \end{bmatrix}$$
Lesson 5: A Slider–Crank Mechanism

\[
\begin{align*}
\dot{\theta}_B &= -\frac{Z_1 \cos \theta_A \dot{\theta}_A}{Z_2 \cos \theta_B} \\
\dot{Z}_3 &= Z_1 \left( \tan \theta_B \cos \theta_A \dot{\theta}_A - \sin \theta_A \dot{\theta}_A \right)
\end{align*}
\]

Hence

\[
\dot{\theta}_B = -\frac{Z_1 \cos \theta_A \dot{\theta}_A}{Z_2 \cos \theta_B} \tag{5.8}
\]

and

\[
\dot{Z}_3 = Z_1 \left( \tan \theta_B \cos \theta_A \dot{\theta}_A - \sin \theta_A \dot{\theta}_A \right) \tag{5.9}
\]

In this example, \(Z_1 = 3\), \(Z_2 = 8\), and the initial conditions are \(\theta_A(0) = \pi/2\) and \(\theta_B(0) = \sin^{-1}(3/8)\).

The solutions can be implemented using a spreadsheet. The Excel spreadsheet file, lesson5.xls, can be found at the publisher’s website. As shown in Figure 5-33, Columns A to I represent time, \(Z_1\), \(Z_2\), \(\dot{\theta}_A\), \(\theta_A\), \(Z_3\), \(\dot{\theta}_B\), \(\dot{Z}_3\), and \(\dot{\theta}_B\), respectively. Note that in this calculation, \(Z_3(0) > 0\) is assumed, hence \(\theta_B(0) < 0\) (clockwise), as illustrated in Figure 5-32. This is consistent with the initial conditions we defined for the motion model.

Figures 5-34 to 5-36 show the graphs of data in Columns F, H, and I, respectively. Comparing Figures 5-34 to 5-36 with Figures 5-12, 5-13, and 5-16, the simulation analysis results are verified.

Note that the angular velocity in Figures 5-16 and 5-36 are in different units. In Figure 5-16, the angular is in degrees/sec. Whereas, in Figure 5-36, it is in rad/sec. After converting the unit to degree/sec, the relative magnitudes (Figure 3-37) matches well. However, the angular velocity is zero at \(t = 0\) in Figure 5-37 and is \(-360\) degree/sec in Figure 5-16. This is because that the angular velocity of the joint Revolute2 reported in Figure 5-16 is a relative measure, which measures the angular velocity of the rod with respect to the crank. When the crank rotates 360 degree/sec, the rod rotates \(-360\) degree/sec.
relatively. In Figure 3-37, the angular velocity \( \omega \), is the angular velocity of the rod referring to the ground. Therefore, it is zero when the crank is in the upright position.

Note that the accelerations of a given joint in the mechanism can be formulated by taking one more derivative of Eqs. 5.5a and 5.5b with respect to time. The resulting two coupled equations can be solved, using Excel spreadsheet. This is left as an exercise.
Exercises:

1. Derive the acceleration equations for the slider-crank mechanism, by taking derivatives of Eqs. 5.5a and 5.5b with respect to time. Solve these equations for the linear acceleration of the piston and the angular acceleration of the joint Revolute2, using a spreadsheet. Compare your solutions with those obtained from COSMOSMotion.

2. Use the same slider-crank model to conduct a static analysis using COSMOSMotion. The static analysis in COSMOSMotion should give you equilibrium configuration(s) of the mechanism due to gravity (turn on the gravity). Show the equilibrium configuration(s) of the mechanism and use the energy method you learned from Sophomore Statics to verify the equilibrium configuration(s).

3. Change the length of the crank from 3 to 5 in. in SolidWorks. Repeat the kinematic analysis discussed in this lesson. In addition, change the crank length in the spreadsheet (Microsoft Excel file, lesson5.xls). Generate position and velocity graphs from both COSMOSMotion and the spreadsheet: Do they agree with each other? Does the maximum slider velocity increase due to a longer crank? Is there any interference occurring in the mechanism?

4. Download five SolidWorks parts from the publisher's web site to your computer (folder name Exercise 5-4).

   (i) Use these five parts, i.e., bearing, crankshaft, connecting rod, piston pin, and piston (see Figure E5-1), to create an assembly like the one shown in Figure E5-2. Note that the crankshaft must orient at 45° CCW, as shown in Figure E5-2.

   (ii) Create a motion model for kinematic analysis. Conduct motion analysis by defining a drive that drives the crankshaft at a constant angular speed of 1,000 rpm.

   (iii) Use the spreadsheet lesson5.xls to calculate the piston velocity. Compare your calculations with those obtained from COSMOSMotion.
6.1 Overview of the Lesson

In this lesson we will discuss how to simulate motion of a spur gear train. A gear train is a set or system of gears arranged to transfer torque or energy from one part of a mechanical system to another. A gear train consists of driving gears that are mounted on the input shaft, driven gears mounted on the output shaft, and idler gears that interpose between the driving and driven gears in order to maintain the direction of the output shaft to be the same as the input shaft or to increase the distance between the drive and driven gears. There are different kinds of gear trains, such as simple gear train, compound gear train, epicyclic gear train, etc., depending on how the gears are shaped and arranged as well as the functions they intend to perform. The gear train we are simulating in this lesson is a compound gear train, in which two or more gears are used to transmit torque or energy. All gears included in this lesson are spur gears; therefore, the shafts that these gears mounted on are in parallel.

In **COSMOSMotion**, gear pair is defined as a special coupler constraint. Joint couplers allow the motion of a revolute, cylindrical, or translational joint to be coupled to the motion of another revolute, cylindrical or translational joint. The two coupled joints may be of the same or different types. For example, a revolute joint may be coupled to a translational joint. The coupled motion may also be of the same or different type. For example, the rotary motion of a revolute joint may be coupled to the rotary motion of a cylindrical joint, or the translational motion of a translational joint may be coupled to the rotary motion of a cylindrical joint. To create a gear pair, we will be coupling two revolute joints. Usually a concentric and a coincident mates will lead to a revolute joint, as seen in previous lessons. Coupling two revolute joints for a gear pair will be carried out in **SolidWorks** using the advance assembly mate option, where two axes that pass through the respective revolute joints (or gears) are picked for the gear mate. The gear mate will be mapped to a gear mate joint in **COSMOSMotion**.

In fact, neither **SolidWorks** nor **COSMOSMotion** cares about the detailed geometry of the gear pair; i.e., if the gear teeth mesh adequately. You may simply use cylinders or disks to represent the gears. No detailed tooth profile is necessary for any of the computations involved. Apparently, force and moment between a pair of teeth in contact will not be calculated in gear train simulations. However, there are other important data being calculated by **COSMOSMotion**, such as reaction force exerting on the driven shaft (for a dynamic analysis), which is critical for mechanism design. In any case, pitch circle diameters are essential for defining gear pair and gear trains in **COSMOSMotion**. Gear ratio of the gear train, which is defined by the ratio of the angular velocities of the output and input gears, is determined by the pitch circle diameters of the individual gear pairs in the gear train.

Although cylinders or disks are sufficient to model gears in **SolidWorks**, we will use more realistic gears throughout this lesson. All gears in the example are shown with detailed geometry, including teeth. In addition, detailed parts, including shafts, bearing, screws and aligning pins are included for a realistic gear train system, as shown in Figure 6-1. In this gear train simulation, we will focus more on graphical
animation, less on computations of physical quantities. We will add a motion driver to drive the input shaft.

### 6.2 The Gear Train Example

**Physical Model**

The gear train example we are using for this lesson is part of a gearbox designed for an experimental lunar rover. The gear train is located in a gear box which is part of the transmission system of the rover, driven by a motor powered by solar energy. The purpose of the gear train is to convert a high-speed rotation and small torque generated by the motor to a low-speed rotation and large torque output in order to drive the wheels of the rover. The gear train consists of four spur gears mounted on three parallel shafts, as shown in Figure 6-1.

![Figure 6-1 The Gear Train System in Rover](image)

The four spur gears form two gear pairs: Pinion 1 and Gear 7, and Pinion 2 and Gear 2, as depicted in Figure 6-1 and 6-2. Note that Pinion 1 is the driving gear that connects to the motion driver; e.g., a motor. The motor rotates in a clockwise direction, therefore, driving Pinion 1. Gear 1 is the driven gear of the first gear pair, which is mounted on the same shaft as Pinion 2. Both rotate in a counterclockwise direction. Gear 2 is driven by Pinion 2, and rotates in a clockwise direction. Note that the diameters of the pitch circles of the four gears are: 50, 120, 60, and 125 mm, respectively; and the numbers of teeth are 25, 60, 24, and 50, respectively. Therefore, the circular pitch \( P_c \), the diametral pitch \( P_d \), and module \( m \) of the first gear pair are, respectively

\[
\begin{align*}
P_c &= \frac{\pi d_{pl}}{N_{pl}} = \frac{\pi(50)}{25} = \frac{\pi(120)}{60} = 6.283 \text{ mm}, \\
P_d &= \frac{N_{pl}}{d_{pl}} = \frac{N_{gl}}{d_{gl}} = \frac{25}{60} = \frac{50}{120} = 0.5 \text{ mm}^{-1}, \\
m &= \frac{d_{pl}}{N_{pl}} = \frac{d_{gl}}{N_{gl}} = \frac{50}{25} = \frac{120}{60} = 2 \text{ mm}.
\end{align*}
\] (6.1)

For the 2nd gear pair, we have
The gear ratio of the gear train shown in Figure 1 is 1:5, i.e., the angular velocity is reduced 5 times at the output. Theoretically, the torque output will increase 5 times if there is no loss due to, e.g., friction. Note that we will use MMGS units system for this lesson.

SolidWorks Parts and Assembly

In this lesson, SolidWorks parts of the gear train have been created for you. There are six files created, gbox_housing.SLDPRT, gbox_input.SLDPRT, gbox_middle.SLDPRT, gbox_output.SLDPRT, Lesson6.SLDASM, and Lesson6withresults.SLDASM. You can find these files at the publisher's web site (http://www.schroffl.com/). We will start with Lesson6.SLDASM, in which the gears are assembled to the housing. In addition, the assembly file Lesson6withresults.SLDASM consists of a complete simulation model with simulation results.

Note that the housing part in SolidWorks was converted directly from Pro/ENGINEER part. The three gear parts in SolidWorks were converted from respective Pro/ENGINEER assemblies. There were nine, nine, and six distinct parts within the three gear assemblies, respectively. These three Pro/ENGINEER assemblies (and associated parts) were first converted to SolidWorks as assemblies. Parts in each assembly were then merged into a single gear part in SolidWorks. The detailed part and assembly conversions as well as merging multiple parts into a single part in SolidWorks can be found in Appendix C.

In the SolidWorks assembly models Lesson6.SLDASM (and Lesson6withresults.SLDASM), there are nine assembly mates. The first three mates, Concentric1, Coincident1, and Coincident2 assemble the input gear to the housing. The input gear is fully constrained. Note that before entering COSMOSMotion, we will suppress Coincident2 in order to allow rotational degree of freedom for the input gear about the Z-axis (see Figures 6-3a, b, and c). Similarly, the next three mates, Concentric2, Coincident4, and Coincident4 assemble the middle gear to the housing. Again, we will suppress Coincident4 to allow the middle gear to rotate about the Z-axis (see Figures 6-3d, e, and f). The third set of mates, Concentric3, Coincident5, and Coincident6 does the same for the output gear (see Figures 6-3g, h, and i). Similarly, Coincident6 will be suppressed to allow for rotation. Note that the three suppressed mates are created to properly orient the three gears, so that the gear teeth mesh well between pairs.
Figure 6-3 Assembly Mates Defined in Lesson6.SLDASM
As mentioned earlier, one important factor for the animation to "look right" is to mesh the gear teeth properly. You may want to use the *Front* view and zoom in to the tooth mesh areas to check if the two pairs of gears mesh well (see Figure 6-4). They should mesh well, which is accomplished by the three coincident mates that will be suppressed before entering *COSMOSMotion*. Note that the tooth profile is represented by straight lines, instead of more popular ones such as involutes, just for simplicity.

![Figure 6-4 Gear Teeth Properly Meshed](image)

If you turn on the axis display (*View > Axes*), axes that pass through the center of the gears about the Z-axis are defined for each gear. These axes are necessary for creating gear mates.

*Simulation Model*

The gear housing will be defined as the ground part. All three gears will rotate with respect to their respective axes. The four gears will be meshed into two gear pairs; *Pinion 1* with *Gear 7*, and *Pinion 2* with *Gear 2*, as discussed earlier. In *SolidWorks*, gear pairs are created by selecting two axes of respective gears (or cylinders) using *Advanced Mates* capability. Before the gear mates can be created, we will suppress the three coincident mates that help properly orient the gears; i.e., *Coincident2*, *Coincident4* and *Coincident6*, in order to allow desired gear rotation motion. When these mates are suppressed, revolute joints will be created between the housing and the three gear parts in *COSMOSMotion*, as shown in Figure 6-5. The revolute joint between the housing and the input gear will be driven at a constant angular velocity of 360 degrees/sec. We will basically conduct a kinematic analysis for this example.
6.3 Using COSMOSMotion

Start SolidWorks and open the assembly file Lesson6.sldasm.

Note that when you open the assembly, you will see a message window, as shown in Figure 6-6, indicating that SolidWorks is unable to locate gbox_input.sldasm. SolidWorks is trying to locate the assembly from which the input gear part was created. Since the input gear assembly and its associated parts are not available in the Lesson 6 folder, SolidWorks is unable to locate it. It is fine to click No and not to locate gbox_input.sldasm. Not locating the assembly file for the input gear part will not affect the motion simulation in this lesson. After clicking No, SolidWorks will ask you to locate the middle gear assembly and output gear assembly. Choose No for both. The assembly files that SolidWorks is looking for are actually located in the subfolder under Lesson 6 as well as the Appendix C folder. You may choose Yes from the message window and locate the missing files in one of these two folders.

Before entering COSMOSMotion there are two things need to be done. First, we will suppress three assembly mates, Coincident2, Coincident4 and Coincident6. Second, we will create two gear mates for the two gear pairs, respectively.

From the Assembly browser, expand the Mates branch, right-click Coincident2, and choose Suppress. The mate Coincident2 will become inactive. Repeat the same to suppress Coincident4 and Coincident6. Save your model.

Next, turn on the axis view by choosing from the pull-down menu, View > Axes. All three axes, one for each gear, will appear in the graphics screen.

We will create two gear mates. Choose from the pull-down menu Insert > Mate. In the Mate dialog box (overlapping with the browser), the Mate Selections field will be active (in red), and is ready for you to pick entities. Pick the axes of the input and middle gears from the graphics screen. Choose Advanced Mates, click Gear, and enter 50mm and 120mm for Ratio, as shown in Figure 6-7. Click the checkmark button on top to accept the mate definition, and click the same button one more time to close the dialog box.

Note that if the axes of the two gears are pointing in the opposite direction, you will have to click Reverse (right below the Ratio text field in the Mate dialog box) to correct the rotation direction. In this example, all three axes are pointing in the same direction. Therefore, do not choose Reverse.

Repeat the same steps to define the second gear mate. This time, pick the axes of the middle and output gears, and enter 60mm and 125mm for Ratio. Two new mates, GearMate1 (gbox_input<g>, gbox_middle<g>) and GearMate2 (gbox_middle<g>, gbox_output<g>), are now listed under Mates.
Now we are ready to enter COSMOSMotion.

Click the Motion button \( \text{enter} \) COSMOSMotion.

Now we are ready to enter COSMOSMotion.

Click the Motion button \# enter COSMOSMotion.

Similar to previous lessons, we will use the browser, and basic drag-and-drop and right-click methods to create and simulate the gear train motion in this lesson.

Before creating any entities, always check the units system. Make sure the units system chosen is MMGS.

Defining Bodies

From the browser, expand the Assembly Components branch. You should see four parts listed, gbox housing-1, gboxinput-1, gboxmiddle-1, and gbox output-1, as shown in Figure 6-8.

Also expand the Parts branch; you should see Moving Parts and Ground Parts listed. Go ahead to move gboxhousing-1 to Ground Parts and move the three gears to Moving Parts by using the drag-and-drop method.

Expand the Constraints branch, and then the Joints branch. You should see two gear mates and three revolute joints listed, as shown in Figure 6-9. In addition, you should see the revolute joint symbols appear in the graphics screen, similar to those of Figure 6-5. Expand all joints in the browser and identify the parts they connect. Take a look at the joint Revolute2 (connecting the input gear to the gear housing), where we will add a driver next. Note that you may see a different revolute joint connecting the input gear to the gear housing. Make sure you pick the right one.

Driving Joint

From the browser, expand the Constraints node and then the Joints node. Right-click the Revolute2 node and choose Properties. In the Edit Mate-Defined Joint dialog box (Figure 6-10), under the Motion tab (default), choose Rotate Z for Motion On, choose Velocity for Motion Type, choose Constant for Function, and enter 360 degrees/sec for Angular Velocity (should appear as defaults). Click Apply to accept the definition. A motion driver symbol should appear at Revolute2, as shown in Figure 6-5.

We are ready to run a simulation. We will use all default simulation parameters.
Running Simulation

Click the Motion Model node, press the right mouse button and select Run Simulation. After a few seconds, you should see the gears start turning. The input gear rotates 360 degrees as expected since the default simulation duration is 7 second.

Saving and Reviewing Results

We will graph the angular velocity of the output gear.

From the browser, expand the Parts branch and then the Moving Parts branch. Right-click the ghox_output-1 node, and choose Plot > Angular Velocity > Z Component (Figure 6-11). The graph should appear and is similar to that of Figure 6-12, which shows that the output velocity is a constant of 72 degrees/sec. Note that this magnitude is one fifth of the input velocity since the gear ratio is 7:5. Both the input (Pinion 7) and output gears (Gear 2) rotate in the same direction. COSMOSMotion gives good results.

Save your model.
Exercises:

1. The same gear train will be used for this exercise. Create a constant torque for the input gear (gboxinput.SLDPRT) about the Z-axis. Turn on friction for all three axles (Steel-Dry/Steel-Dry). Define and run a 2-second dynamic simulation for the gear train.

   (i) What is the minimum torque that is required to rotate the input gear, and therefore, the entire gear train?

   (ii) If the torque applied to the input gear is 100 mm N, what is the output angular velocity of the gear train at the end of the 2-second simulation? Verify the simulation result using your own calculation.

   (iii) Create a graph for the reaction moment between gears of the first gear pair (GearMatel) due to the 100 mm N torque. What is the reaction moment obtained from simulation?
Notes:
7.1 Overview of the Lesson

In this lesson, we will learn cam and follower, or cam-follower. A cam-follower is a device for converting rotary motion into linear motion. The simplest form of a cam is a rotating disc with a variable radius, so that its profile is not circular but oval or egg-shaped. When the disc rotates, its edge (or side face) pushes against a follower (or cam follower), which may be a small wheel at the end of a lever or the end of the lever or rod itself. The follower will thus rise and fall at exactly the same amount as the variation in radius. By profiling a cam appropriately, a desired cyclic pattern of straight-line motion, in terms of position, velocity, and acceleration, can be produced.

We will learn to create a motion model and simulate the control of opening and closing of an inlet or exhaustive valve, usually found in internal combustion engines, using cam-follower connections. In a design such as that of Figure 7-1, the drive for the camshaft is taken from the crankshaft through a timing chain, which keeps the cams synchronized with the movement of the piston so that the valves are opened or closed at a precise instant. The mechanism we will be working with consists of bushings, camshaft, pushrod, rocker, valve, valve guide and spring, as shown in Figure 7-1. The cam-follower connects the camshaft and the pushrod. When the cam on the camshaft pushes the pushrod up, the rocker rotates and pushes the valve on the other side downward. The spring surrounding the valve gets compressed, and opens up the inlet for air to flow into the combustion chamber.

7.2 The Cam and Follower Example

Physical Model

The camshaft and the rocker will rotate about the axes of their respective revolute joints connecting them to their respective bearings (defined as ground body). The camshaft is driven by a motor of constant velocity of 600 rpm (or 10 rev/sec). The profile of the cam consists of two circular arcs of 0.25 and 0.5 in. radii, respectively, as shown in Figure 7-2. The lower arc is concentric with the shaft, and the center of the upper arc is 0.52 in. above the center of the shaft. When the camshaft rotates, the cam mounted on the shaft pushes the pushrod up by up to 0.27 in. (that is, 0.52+0.25-0.5 = 0.27). As a result, the rocker will rotate and push the valve at the other end downward at a frequency of 10 times/sec. The valve will move again up to 0.27 in. downward since the pushrod and the valve are positioned at an equal distance from the rotation axis of the rocker. When the camshaft rotates where the larger circular arc (0.5 in. radius) of
the cam is in contact with the follower (in this example, the pushrod), the pushrod has room to move downward. At this point, the rocker will rotate back since the spring is being uncomprssed. As a result, the valve will move up, and therefore, close the inlet. The valve will be open for about 120 degree per cycle, based on the cam design shown in Figure 7-2.

The unit system chosen for this example is IPS and all parts are made up of steel.

SolidWorks Parts and Assembly

The cam-follower system consists of seven parts, bushing (two), valve guide, camshaft, pushrod, rocker, and valve, as shown in Figure 7-1. In addition, there are two assembly files, Lesson7.SLADM and Lesson7withresults.SLADM that you may download from the publisher’s web site.

We will start with Lesson 7.SLADM, in which the parts are adequately assembled. In this assembly the first bushing is anchored (ground) and the second bushing and the valve guide are fully constrained. These three parts will be assigned as ground parts. The remaining four parts will be defined as movable parts in COSMOSMotion.

Same as before, the assembly file Lesson7withresults.SLADM contains a complete simulation model with simulation results. You may want to open the assembly to see the motion animation of the mechanism. In the assembly where a motion model is completely defined, a mate has been suppressed to allow movement between components. You can also see how the parts move by right clicking in the graphics screen, choosing Move Component, and dragging any movable parts; for example the camshaft to rotate with respect to the second bushing. The whole mechanism will move accordingly.

There are eighteen assembly mates, including four coincident, three concentric, seven distance, two parallel, one tangent, and one cam-mate-tangent, as listed in the browser (see Figure 7-3). You may want to expand the Mates branch in the browser to review the list of assembly mates. Move the cursor over any of the mates; you should see the entities selected for the assembly mate highlighted in the graphics screen.

As mentioned earlier, the first bushing, bushing<1>, shown in the browser is anchored to the assembly. The second bushing (bushing<2>) and the valve guide (valve guide<1>) were fully assembled to bushing<1>. The first three mates, Coincident 1, Distance 1, and Distance 2, were employed to assemble bushing<2> to bushing<1>. And the next three distance mates assemble the valve guide to bushing<1>.
Note that the distance mates are essentially coincident mate with distance between entities. The distance mates were created to properly position the second bushing with respect to the first bushing. All three parts will be defined as the ground part in COSMOSMotion.

The next two mates, Concentric1 and Coincident1, assemble the rocker (rocker1) to the first bushing (bushing1), allowing a rotation degree of freedom about the Z-axis, as shown in Figures 7-4a and b. COSMOSMotion will map a revolute joint between the rocker and the first bushing.

The next part assembled is the pushrod (pushrod1). The pushrod was assembled to the rocker using Tangent1, Distance6, and Coincident3 mates, as shown in Figures 7-4c, d, and e, respectively. As a result, the pushrod is allowed to move vertically at a distance of 1.25 in. from the Right plane of the first bushing (Distanced), at the same time, maintaining tangency between the top of the cylindrical surface of the pushrod and the socket surface of the rocker.
The next part is the camshaft \((\text{cam\_shaft}<1>)\). The camshaft was first assembled to the second bushing \((\text{bushing}<2>)\), and then to the pushrod. \textit{Concentric2} aligns the camshaft and the second bushing. \textit{Coincident4} mates the center plane of the camshaft to that of pushrod, as shown in Figures 7-4f and g. As a result, the camshaft is allowed to rotate about the \(Z\)-axis.

In addition, the \textit{CamMateTangent1} defines a cam and follower between the cam surface of the camshaft and the cylindrical surface at the bottom of the pushrod. Note that all surrounding surfaces on the cam and follower must be selected for the cam-follower joint. In this case, four surfaces are selected for the cam (mounted on the camshaft) and one surface is included on the follower, as shown in Figure 7-4h. Note that the assembly should have overall one degree of freedom at this point. You may either rotate the camshaft, the rocker, or move the pushrod vertically to see the relative motion of the assembly. Use
Apparently, this result implies that there are redundant dofs created in the system. This is fine since COSMOSMotion filters out the redundant dofs. You may want to check the redundancy by choosing, from the pull-down menu, COSMOSMotion > Show Simulation Panel, as discussed in Appendix A. You may want to review Appendix A for more information about defining joints and calculating degrees of freedom.

The final motion model is shown in Figure 7-6, where the Z-rotation of the concentric joint, Concentric2, between the camshaft and the second bushing is driven by a constant angular velocity of
3,600 degrees/sec; i.e., 600 rpm, about the Z-axis of the global coordinate system. In addition, a spring surrounding the valve will be created in order to provide a vertical force to push the rocker up, therefore, close the valve. The spring has a spring constant of 10 lbf/in and an unstretched length of 1.25 in. The spring is created between the bottom face of the rocker and top face of the valve guide.

7.3 Using COSMOSMotion

Start SolidWorks and open assembly file Lesson 7. SLDASM.

From the browser, click the Motion button if to enter COSMOSMotion.

Again, always check the units system. Make sure that IPS units system is chosen for this example.

Defining Bodies

From the browser, expand the Assembly Components branch. You should see seven entities listed, Bushing-1, Bushing-2, cam_shaft-1, pushrod-1, rocker-7, valve guide-1, and valve-1, as shown in Figure 7-7. Also expand the Parts branch; you should see Moving Parts and Ground Parts listed. We will move Bushing-1, Bushing-2, and valve guide-1 to Ground Parts and the remaining four parts to Moving Parts by using the drag-and-drop method.

From the browser, click Bushing-1. Press the Ctrl key and click Bushing-2 and valve guide-1. All three parts should be selected. Drag and drop them to the Ground Parts node.

Repeat the same to select the remaining four parts under the Assembly Component branch. Drag and drop them to the Moving Parts node.

Expand the Constraints branch, and then the Joints branch. You should see that ten joints are listed (see Figure 7-5). All joint symbols should appear in the graphics screen, similar to that of Figure 7-6. Expand all joints in the browser and identify the parts they connect. Take a look at the joint Coincident2 (connecting camshaft to Bushing-2), where we will add a driver next.
Driving Joint

Right click the Concentric2 node and choose Properties (see Figure 7-8). In the Edit Mate-Defined Joint dialog box (Figure 7-9), under the Motion tab (default), choose Rotate Z for Motion On, choose Velocity for Motion Type, choose Constant for Function, and enter 3600 degrees/sec for Angular Velocity, as shown in Figure 7-9. Click Apply to accept the definition.

Defining Spring

From the browser, expand the Forces branch, right click the Spring node and choose Add Translational Spring (see Figure 7-10). In the Input Spring dialog box (Figure 7-11), the Select 1st Component field should be highlighted in red and ready for you to pick. Rotate the view and pick the bottom face of the rocker (see Figure 7-12), the Select 2nd Component field should now highlight in red, and rocker-1/DDMFace19 should appear in the Select Point on 1st Component field, which indicates that the spring will be connected to the center point of the face selected.

In case you picked a wrong entity, simply select the entire text in the respective text field in the dialog box, and press the Delete key to delete the text. The text field will turn back to red and will be ready for you to pick another entity.

Rotate the view back, and then pick the top face in the valve guide, as shown in Figure 7-12. Now, valve guide-1 and valve guide-1/DDMFace20 should appear in the Select 2nd Component field and Select Point on 2nd Component field, respectively. Also, a spring should appear in the graphics screen, connecting the center points of the two faces.

Enter the followings:

Stiffness: 10
Length: 1.25 (Note that you have to deselect the Design box to the right before entering this value)
Force: 0
Coil Diameter: 0.75
Number of coils: 8
Wire Diameter: 0.1

Click Apply to accept the spring definition and close the Insert Spring dialog box.

Defining and Running Simulation

Click the Motion Model node, press the right mouse button and select Simulation Parameters. Enter 0.5 for simulation duration and 500 for the number of frames.

Click the Motion Model node again, press the right mouse button and select Run Simulation. You should see that the camshaft starts rotating, the pushrod is moving up and down, which drives the rocker,
and then the valve. The camshaft rotates 5 times in the 0.5-second simulation duration. We will graph the position, velocity, and acceleration of the valve next.

Graph the 7-velocity and 7-acceleration of the valve by choosing Plot > CM Velocity (and CM Acceleration) > Y Component. The graphs of the velocity and acceleration are shown in Figures 7-15 and 16, respectively. As shown in Figure 7-15, there are two velocity spikes per cycle, representing that the valve is pushed downward (negative velocity) for opening and is being pulled back (positive velocity) for closing, respectively. The valve stays closed with zero velocity.

Figure 7-16 reveals high accelerations when the valve is pushed and pulled. Note that such a high acceleration is due to high-speed rotation at the camshaft. This high acceleration could produce large inertial force on the valve, yielding high contact force between the top of the valve and the socket surface in the rocker. We would like to check the reaction force between the top of the valve and the rocker. The

**Displaying Simulation Results**

From the browser, expand the Parts branch, and then the Moving Parts branch. Right click the valve-1 node, and choose Plot > CM Position > Y (see Figure 7-13). The graph of the Y-position of the mass center of the valve will appear, similar to that of Figure 7-14, where the valve is moving between −1.96 and −1.70 in., traveling about 0.26 in., which is about what was expected, as discussed in Section 7.2.

As shown in Figure 7-14, the flat portion on top indicates that the valve stays completely closed, which spans about 0.066 seconds, approximately 240 degrees of the camshaft rotation in a complete cycle. Therefore, the valve will open for about 0.034 seconds per cycle, roughly 120 degrees.

Graph the 7-velocity and 7-acceleration of the valve by choosing Plot > CM Velocity (and CM Acceleration) > Y Component. The graphs of the velocity and acceleration are shown in Figures 7-15 and 16, respectively. As shown in Figure 7-15, there are two velocity spikes per cycle, representing that the valve is pushed downward (negative velocity) for opening and is being pulled back (positive velocity) for closing, respectively. The valve stays closed with zero velocity.

Figure 7-16 reveals high accelerations when the valve is pushed and pulled. Note that such a high acceleration is due to high-speed rotation at the camshaft. This high acceleration could produce large inertial force on the valve, yielding high contact force between the top of the valve and the socket surface in the rocker. We would like to check the reaction force between the top of the valve and the rocker. The
graph of the reaction force can be created by expanding Constraints and Joints branches, right clicking Concentric2 (between the valve and the rocker), and choosing Plot > Reaction Force > Y Component.

The reaction force graph (Figure 7-17) shows that the reaction force between the top of the valve and the socket face of the rocker is about 0.4 lb., which is insignificant. Note that this small reaction force can be attributed to the small mass of the valve. If you open the valve part and acquire its mass (from pull-down menu, choose Tools > Mass Properties), the mass of the valve is 0.03 lb. Therefore, the inertia for the valve at the peak accelerations is about $0.03 \times 3500 \text{ in/sec}^2 = 105 \text{ lb. in/sec}^2 = 168/386 \text{ lb} = 0.44 \text{ lbf}$, which is consistent to peaks found in Figure 7-17. If you are not quite sure about why this 386 is factored in for force calculation, please refer to Appendix B for mass and force unit conversions. Save your model.

![Figure 7-15 Graph of Valve Velocity](image1)

![Figure 7-16 Graph of Valve Acceleration](image2)

![Figure 7-17 Graph of the Reaction Force](image3)
Exercises:

1. Redesign the cam by reducing the small arc radius from 0.25 to 0.2 and reducing the center distance of the small arc from 0.52 to 0.40, as shown in Figure E7-1. Repeat the dynamic analysis and check reaction force between the valve and the rocker. Does this redesigned cam alter the reaction force?

2. If we change the Parallel! mate between the Right plane of the valve and the Right plane of the first bushing to a distance mate, will the mechanism move? What other changes must be made in order to create a valid and movable mechanism similar to that was presented in this lesson?
8.1 Overview of the Lesson

This is an application lesson. We will apply what we learned in previous lessons to a real-world application. This application involves designing a device that can be mounted on a wheelchair to mimic soccer ball-kicking action while being operated by a child sitting on the wheelchair with limited mobility and hand strength. Such a device will provide more incentive and realistic experience for children with physical disabilities to participate in soccer games. This example was extracted from an undergraduate student design project that was carried out in conjunction with a local children hospital. This device was intended primarily to be used in the summer camp sponsored by the children hospital.

The focus of this lesson is slightly different from previous ones. Instead of focusing on discussing how to use COSMOSMotion to create motion entities, we will focus on how to use COSMOSMotion to support design. In order to narrow down the design options to be more manageable, we will assume all major components are designed with dimensions determined. More specifically, we will use COSMOSMotion to help choose a spring, as well as determine if the required operating force is acceptable. Since the users of this device are children with limited physical hand strength, the operating force must be minimized in order to make the device useful.

The examples we have discussed in previous lessons are simple enough so that some of the simulation results can be verified by hand calculations (for example, using a spreadsheet). However, most of the real-world applications, including the example of this lesson, are too complicated to verify by hand calculations. When we are dealing with such applications, two principles are helpful in leading to successful simulations. First, the simulation model you created has to be physically meaningful and as consistent to the physical conditions as possible. Second, very often you will have to make assumptions in order to simplify the problems so that the simulations can be carried out. This is because that the simulation models must comply with the capabilities of the software you are using. In order to effectively use the simulations to support design, you will have to understand the physical problems very well; in the mean time, be familiar with the capabilities and limitations of the software you are using.

Since most of you will be often learning the software as you are tackling simulation and/or design problems, it is strongly recommended that you employ the principle of spiral development to incrementally build up your simulation model. In another word, you may want to start from a simplified model with simple scenarios by making adequate assumptions to your simulation model. Make the simplified model works first, then relax the assumptions and add motion entities to make your model closer to the real situation. Repeat the process until you reach a simulation model and simulation scenarios that answer your questions and help you make design decisions. In each step, make sure that the simulation model does what you expect it to do before bringing it to the next level.

In this lesson, we will employ the spiral development principle. We will assume that all components and their physical dimensions are determined. The design is essentially narrowed down to the selection of
a spring, including both spring constant and free length, and to ensure that the required force is small enough for a child to easily operate the device. We will try our best to check and hopefully verify the simulation model in each step. Note that COSMOSMotion is very sensitive to some of the conditions and parameters, such as the initial condition (that is, the handle bar orientation), spring constant, etc. Some of the conditions simulated are physically meaningful and yet COSMOSMotion gives unrealistic simulation results due to its limitations. Again, COSMOSMotion is not foolproof. One cannot blindly accept the simulation results. When a result is determined unrealistic after reviewing animation, graphs, etc., the best way to proceed is to compose a simpler simulation model and/or try a different (more idealized) scenario until the simulation result is physically meaningful based on your educated judgment. You will see trials-and-errors in this lesson and many other real-world applications in the future.

8.2 The Assistive Device

Physical Model

This assistive device for soccer games consists of five major components: the clamper, handle bar, plate, kicking-rod, and spring, as illustrated in Figure 8-1. In reality these five components will be assembled first and clamped to the lower frame of the wheelchair for use.

The handle bar is mounted to the plate at the pivot pin of the plate and linked to the middle pin of the kicking rod. The kicking rod is inserted into the two lower brackets mounted on the plate. When the handle (on top of the handle bar) is pulled backward, the handle bar rotates about the pivot pin; therefore, drives the kicking rod to move forward along the longitudinal direction through the link between the bottom slot of the handle bar and the middle pin of the kicking rod. The forward movement of the kicking rod produces momentum to "kick" the soccer ball. A spring is added between the upper bracket and the handle bar to restore the handle bar to its neutral position after pulling. The spring also helps the user to pull the handle bar with a lesser force.
The focus of this lesson is to use COSMOSMotion to simulate the position and velocity of the kicking rod for a given force that can be comfortably provided by a child with limited physical strength. Lots of factors contribute to the operating force of the mechanism. For example, one of the critical parameters is the location of the pivot pin. The lower the pivot pin is located the lesser force is required to operate the mechanism. However, the purpose of this lesson is not necessarily to determine the final design of the device, but illustrate the process of using COSMOSMotion to assist the design. Therefore, as mentioned earlier, the scope of the design has been narrowed down to the selection of the spring and to determine if the force is small enough for a child to operate the device.

SolidWorks Parts and Assembly

The assembly of the mechanism consists of eight parts and one subassembly. These parts are handle.SLDPRT, plate.SLDPRT, rod.SLDPRT, foot.SLDPRT, clamper.SLDPRT, joint.SLDPRT, collar.SLDPRT, and wheelchair.SLDPRT. The subassembly is kickingrod.SLDASM which consists of rod.SLDPRT and foot.SLDPRT.

In addition, there are six assembly files, Lesson8.SLDASM, Lesson8TaskOne.SLDASM, Lesson8TaskTwo.SLDASM, Lesson8TaskThreeNoFriction.SLDASM, Lesson8TaskThreeSmallFriction.SLDASM, and Lesson8TaskThreeLargeFriction.SLDASM. You can download these files from publisher’s web site.

Same as before, the assembly files, with the exception of Lesson8.SLDASM, consist of complete simulation models with simulation results under respective simulation scenarios. You may want to open these files to see the motion animations of the mechanism. In these assembly files, a mate (Angle) has been suppressed. You can also see how the parts move by right clicking in the graphics screen, choosing Move Component, and dragging any movable parts; for example dragging the handle to drive the kicking rod.

There are fifteen assembly mates, including four coincident, three concentric, four distance, two parallel, and one angle, defined in the assembly, as listed in the browser (see Figure 8-2). You may want to expand the Mates branch in the browser to see these assembly mates. Move your cursor over any of the mates; you should see the entities chosen for the assembly mate highlighted in the graphics screen.

The first nine mates assemble the clamper to the wheelchair, the joint to the clamper and the wheelchair, and then the plate to the joint part. Note that the joint is a SolidWorks part that connects the plate and the clamper rigidly. These nine mates are pretty standard. All these parts are fully constrained and are fixed to the wheelchair.

The next two mates, Concentric3 and Distance4, assemble the handle bar to the plate at the pivot pin, allowing the handle bar to rotate at the pivot pin, as shown in Figure 8-3a. COSMOSMotion will convert these mates to a revolute joint. The distance mate provides an adequate clearance between the handle bar and the plate, as shown in Figure 8-3b.
Figure 8-3 Assembly Mates Defined in Lesson8.SLDASM
The next three mates, CamMateTangent1, Coincident3, and Coincident4, assemble the kicking rod to the plate and the handle bar. First, the middle pin of the kicking rod is assembled to the inner surface of the bottom slot of the handle bar using CamMateTangent1, as shown in Figure 8-3c. As a result, the pin can only move within the slot, which is desirable. Second Coincident3 mates the bottom face of the kicking rod to the inner bottom face of the first lower bracket of the plate, as shown in Figure 8-3d. Similarly Coincident4 mates the rear face of the kicking rod to the inner side face of the first lower bracket of the plate, as shown in Figure 8-3e. These two mates restrict the kicking rod to slide along the longitudinal direction, which will be converted to a translational joint by COSMOSMotion.

The final mate, Anglel, orients the vertical plane of the handle bar (Right Plane) respect to that of the plate (Right Plane). This mate will help determine an initial condition for motion simulations. Note that the Anglel mate to orient the handle, it will has to be suppressed to allow the handle bar to rotate.

Figure 8-4 Travel Distances of the Kicking Rod and the Handle Bar
Based on the geometry of the mechanism and dimensions of its constituent components, the kicking rod is able to travel a total of 12.2 in. along the longitudinal direction (Z-direction). The kicking rod can move 6.82 in. forward (positive Z-direction with respect to the middle pin) until its middle pin becomes in contact with the inner face at the lower end of the slot, as shown in Figure 8-4a. Similarly, the kicking rod moves 5.34 in. backward until the side face of the handle bar becomes in contact with the front end face of the first lower bracket, as shown in Figure 8-4b.

At the same time, the handle will travel a total of 18.6 in., 9.98 backward and 8.6 in. forward, as shown in Figures 8-4c and 8-4d, respectively. Note that these measurements shown in Figure 8-4 can be obtained by using the Measure option in SolidWorks. To access the Measure option, simply choose from the pull-down menu Tools > Measure.

Simulation Model

In this motion model, the only moving parts are the handle bar and the kicking rod. After suppressing the mate Angle1, COSMOSMotion will add a revolute joint to the motion model between the handle bar and the plate at the pivot pin, as shown in Figure 8-5. The handle bar is allowed to rotate about the X-axis of the global coordinate system at the pivot pin. In addition, a translational joint is created by COSMOSMotion between the kicking rod and the first lower bracket. This joint constrains the kicking rod to translate along the longitudinal direction; i.e., the Z-direction. The third and final joint is the CamMateTangent between the outer surface of the middle pin and the inner surface of the slot at the bottom of the handle bar. This joint restricts the pin to move inside the slot.

In addition to joints, a spring is added between the upper bracket of the plate and the hook on the side of the handle bar. This spring, as shown in Figure 8-5, is added to restore the handle bar to a neutral position after pulling. Since the free spring length is set to a slightly larger value so that the handle bar will be oriented at a negative 5~10-degree angle (about the X-axis). Therefore, the user will have to push the handle forward before pulling it back to kick the ball. A larger spring free length will push the handle leaning further backward (toward the user), providing additional pulling force that helps the users to pull back the handle bar. Note that the pulling action will push the kicking rod forward (positive Z-direction) to "kick" the soccer ball.

There is one contact constraint added to the motion model. The contact constraint is defined between the outer side face of the handle bar and the front end face of the first lower bracket. This contact constraint will prevent the handle bar from moving further backward and penetrating through the brackets. In this contact constraint, a restitution coefficient of 0.5 is assumed.

Finally, an impulse force of 5-65 lb in a time span of 1.0 second will be added to the handle bar along the Z-direction, as shown in Figure 8-5, to simulate the operating force. Note that the impulse force must first push the handle bar about 10 degrees forward before pulling it back, as illustrated in Figure 8-6.
In this example, we will turn on the gravity, which is acting in the negative 7-direction (vertically downward); i.e., the default setting. We will first assume no friction in any joints. This assumption will greatly simplify the simulation model and help set up the motion model correctly. We will use this model to choose a spring, including the selection of spring constant and free length. The simulation results of this non-friction model have been created in the assembly files Lesson8TaskOne.SLDASM and Lesson8TaskTwo.SLDASM.

The friction force will be turned on for both the revolute and translational joints to determine the required force for operating the mechanism. In addition, the operating force, modeled as an impulse force will be added to the mechanism. Note that the friction is not added to the CamMateTangent joint between the middle pin and the slot since such a capability is not currently supported by COSMOSMotion. Results of these simulations can be found in Lesson8TaskThreeNoFriction.SLDASM, Lesson8TaskThreeSmallFriction.SLDASM, and Lesson8TaskThreeLargeFriction.SLDASM.

8.3 Using COSMOSMotion

In this example, we will start with a valid simulation model defined in the assembly Lesson8.SLDASM. The unit system is IPS. When you open this assembly file, you should see a properly assembled model with fifteen mates, as shown in Figure 8-2, where the last constraint, Angle1, should have been suppressed. Note that the mate angle is set to positive 10 degrees for the time being, and the handle bar is leaning forward (toward the Z-direction), as shown in Figure 8-7. Note that this angle is what we assume for Task One simulations.

If you enter COSMOSMotion and expand the Parts and Constraints branches in the browser, you should see the existing motion entities, as shown in Figure 8-8. There are four parts under the Ground Parts branch. Only handle bar and kicking rod are movable. In addition, there are three joints defined, CamMateTangent, Revolute, and Translational, as discussed earlier. In the Contact branch under the Constraints, there is one contact joint, Contact 3D, defined to prevent the kicking rod from penetrating into the brackets. Due to the contact constraint and the restitution coefficient defined, the handle bar will bounce back when it hits the front edge of the first lower bracket.
There are three major tasks we will carry out in this design. In Task One we will run simulation to check the function of the 3D contact constraint. We hope to ensure that all joints are adequately defined, and the restitution coefficient we assumed in the contact constraint gives us reasonable results.

In Task Two, we will add the spring. We will adjust the spring constant and its free length until we reach an equilibrium configuration that we can work with. Note that a desired free length should bring the handle bar backward a negative 5-10 degree angle (about the X-axis). After the spring is determined, we will add an operating force at the handle in Task Three. Note that the force will first push the handle bar forward about 10 degrees (positive, about the X-axis) before pulling it backward in order to provide enough travel distance for the kicking rod, and hopefully, sufficient momentum to kick the ball.

In Task Three, we will start with a non-friction case, and then turn on friction in both the revolute and translational joints. From the friction cases, we will determine if the operating force is sufficient to push the handle forward about 10 degrees before pulling it backward. We will vary the friction coefficients to simulate different scenarios and then determine the magnitude of the operating forces accordingly. The magnitude of the force will answer the critical question: if the design is acceptable. We hope to keep the maximum force magnitude under 20 lbf.

Task One: Determining Contact Constraints

We will carry out simulations for the motion model defined in Lesson8.SLDASM. In the assembly, the handle is leaning forward 10 degrees (see Figure 8-7), the contact joint Contact 3D is defined, and the gravity is turned on. You may want to open the Contact3D constraint by right clicking Contact3D from the browser and choosing Properties. In the Edit 3D Contact dialog box, you should see that the plate and handle are included in the first and second containers, respectively, as shown in Figure 8-9. If you choose the Contact tab, you should see that the Coefficient of Restitution is 0.5, as shown in Figure 8-10. Note that no friction is imposed for this constraint. Close the dialog box by clicking the Apply button.
Also, we would like to make sure that the gravity is set up properly. From the browser, right click the Motion Model node and select System Defaults. In the Options dialog box (Figure 8-11), you should see the acceleration is 386.22 in/sec^2, and the Direction is set to -1 for Y. Click OK to accept the gravity setting.

Click the Motion Model node from the browser, press the right mouse button and select Simulation Parameters. Enter 2 for simulation duration and the 200 for the number of frames, as shown in Figure 8-
12. Make sure the *Use Precise Geometry for 3D Contacts* is selected in order for *COSMOSMotion* to detect contact during simulation.

Click the *Motion Model* node again, press the right mouse button and select *Run Simulation*. You should see that the handle bar start moving forward and the kicking rod moving backward due to gravity. When the handle bar and the first lower bracket is in contact, the handle bar bounces back slightly due to the contact constraint we defined.

Next, we will graph the position and velocity of the kicking rod along the Z-direction.

From the browser, expand the *Parts* node and then the *Moving Parts* node. Right click *kickingrod-7*, and choose *Plot > CM Position > Z*, and *Plot > CM Velocity > Z Component*.

Two graphs like those of Figures 8-13 and 8-14 should appear. Note that the vertical scales of the graphs have been adjusted for clarity. The position graph shows that the mass center of the kicking rod was located at $Z = 21.6$ in. initially. The mass center moves to the right to $Z = 17.8$ in., where the handle bar is in contact with the first lower bracket. The handle bar, therefore the kicking rod, bounces back, and the center mass of the kicking rod reaches to $Z = 18.6$ in. before it slides to the right again due to gravity. After about 1.5 seconds, the kicking rod rests and stays in contact with the bracket.

The velocity graph (Figure 8-14) shows that the bouncing velocity is half of the incoming velocity in the opposite direction. This is certainly due to the 0.5 restitution coefficient defined at the contact constraint.

From these two graphs, we conclude that the motion model has been defined correctly. Save your model. You may want to save the model under different name and use it for Task Two simulations.

**Figure 8-13  CM Position of the Kicking Rod**

<table>
<thead>
<tr>
<th>Time (sec)</th>
<th>Position (in)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>22.0</td>
</tr>
<tr>
<td>0.5</td>
<td>21.0</td>
</tr>
<tr>
<td>1.0</td>
<td>20.0</td>
</tr>
<tr>
<td>1.5</td>
<td>19.0</td>
</tr>
<tr>
<td>2.0</td>
<td>18.0</td>
</tr>
</tbody>
</table>

**Figure 8-14  CM Velocity of the Kicking Rod**

<table>
<thead>
<tr>
<th>Time (sec)</th>
<th>Velocity (in/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>-5.0</td>
</tr>
<tr>
<td>0.5</td>
<td>-4.5</td>
</tr>
<tr>
<td>1.0</td>
<td>-4.0</td>
</tr>
<tr>
<td>1.5</td>
<td>-3.5</td>
</tr>
<tr>
<td>2.0</td>
<td>-3.0</td>
</tr>
</tbody>
</table>

**Task Two: Adding Spring**

In Task Two, we will add a spring to the mechanism. We will adjust the spring constant and its free length until we reach an equilibrium configuration that we can work with. Note that the free length we specify will have to bring the handle bar backward about 5-10 degrees (that is, negative 5-10 degrees about the X-axis).
Delete the simulation result.

From the browser, right click the Spring node and choose Add Translational Spring, the Insert Spring dialog box will appear (Figure 8-15). Pick the center hole of the upper bracket and the hook of the handle bar, as shown in Figure 8-16. Enter the followings:

- **Stiffness**: 30
- **Length**: 5
- **Force**: 0
- **Coil Diameter**: 1
- **Number of coils**: 10
- **Wire Diameter**: 0.25

Note that the actual distance between the hole and the hook under the current configuration is about 4.34 in., which can be obtained by clicking the Design box to the right of the Length field. The number we enter, 5, is larger than the actual distance. Again, the purpose of entering a larger free spring length is to make the handle bar lean backward at equilibrium.

Click the Apply button to accept the spring. In the current configuration, the spring is compressed since the handle is leaning 10 degrees forward.

Run a simulation.

You should see that the handle bar start moving backward and the kicking rod moving forward due to the stretching of the spring.

When the simulation is completed, the position and velocity graphs of the kicking rod will appear, similar to Figures 8-17 and 8-18. Both the position and velocity graphs reveal a sinusoidal type curve. This is due to the fact that no friction has been applied to the joints. Also, the handle bar is not colliding with the bracket. The graphs show that the period of one vibration is just under 0.5 seconds.
Now we will add a graph to show the rotation angle of the handle bar. Delete the simulation result. From the browser, right click the handle-1 and choose Plot > Bryant Angles > Angle7; i.e., about the X-axis. Rerun the simulation, the angle graph should appear, similar to Figure 8-19. The graph shows that the handle bar is oscillating between 10 and -9 degrees. The angle of the handle bar at equilibrium (assuming friction is added to dissipate energy) will be roughly the average of 10 and -9; i.e., less than 1 degree, which is far less than the desired angle (negative 5-10 degrees).

Delete the simulation result, change the free length to 5.5 in., and rerun the simulation. The angle graph shown in Figure 8-20 indicates that the handle bar oscillates between 10 and -25 degrees, and the mean angle is about -7 degrees, which is desired.

Note that a softer spring will increase the oscillation angle. If the spring constant is too small, say 2 lbf/in, the handle bar may reach a deadlock position with the middle pin of the kicking rod, similar to what was shown in Figure 8-4a, which is not desirable. On the other hand, if the spring is too stiff, the
force required to push the handle bar forward may be excessive. Currently, the spring constant is set to 30
lbf/in. This parameter will be revisited later in Task Three. Save your model before moving to Task Three.

Task Three: Determining the Operating Force

In Task Three, we will determine the operating force. The force must be small enough to allow children with limited physical strength to operate the mechanism. We will add a force at the handle, and use simulation results to determine the required operating force.

The force to be determined is supposed to push the handle forward about 10 degrees, as shown in Figure 8-6, and then pull it backward to push out the kicking rod. Then, we will turn on friction at both translational and revolute joints, and determine the operating force again. We will adjust the friction coefficient (assuming different physical conditions) in order to determine a range of the operating force. In the simulation, we will start with a configuration where the handle bar is set to -7 degrees; i.e., in its neutral position, as determined in Task Two.

Save the model under a different name, say Task3. Delete the simulation result, and go back to SolidWorks assembly mode by clicking the Assembly buttons on top of the browser.

First unsuppress the mate AngleL. Then, right click the AngleL mate and choose Edit Feature. In the AngleL window (Figure 8-21), change the angle to 7, click the Flip direction button (to deselect it), and click the checkmark button on top to accept the definition. The handle bar should rotate to a 7-degree position backward in the graphics screen, as shown in Figure 8-22. This is the initial configuration for Task Three simulations.

Next we will create a force at the handle.

The force can be added from the browser by expanding the Forces branch, right clicking the Action Only node, and choosing Add Action-Only Force. In the Insert Action-Only Force dialog box, the Select Component to which Force is Applied field (see Figure 8-23) is active (highlighted in red) and ready for you to pick the entities.

Rotate the model and pick the face of the handle, as shown in Figure 8-24. The part handle-1 is now listed in the Select Component to which Force is Applied field, and handle-1/DDMFace 10 is listed in both the Select Location and the Select Direction fields.
Now the Select Reference Component to orient Force field is active for selection. We will click the ground button to the right of the field. After that, you should see Assem5 (representing the ground) appear in the text field. A force symbol will appear in the graphics screen pointing downward, which is not what we want. We will have to change the force direction.

Now the Select Reference Component to orient Force field is active for selection. We will click the ground button to the right of the field. After that, you should see Assem5 (representing the ground) appear in the text field. A force symbol will appear in the graphics screen pointing downward, which is not what we want. We will have to change the force direction.

Select all the text in the Select Direction field (press the let mouse button and drag to select all text), and press the Delete key to delete the current selection. Pick the end face of the foot, as shown in Figure 8-24, to orient the force along the negative Z-direction. The arrow of the force symbol should now point to the negative Z-direction, which is normal to the face we picked.

The maximum force is 5 lb. Note that the negative sign for the first few force data is to reverse the force direction in order to push the handle bar forward.
Click the graph button (right most, as circled in Figure 8-25), the function graph will appear like the one in Figure 8-26. Note that internally COSMOSMotion will create a smooth spline function using the data entered.

Close the graph and click Apply button to accept the force definition.

Run a simulation. The result graphs will appear at the end of the 2-second simulation.

The angle graph (Figure 8-27) shows that the handle bar starts at an orientation angle of -7 degrees as expected, and swing forward to a 13-degree angle, which is desirable, due to the forward force. The handle then moves backward to about 26 degrees, and then oscillates.

The position graph in Figure 8-28 shows that the mass center of the kicking rod starts at about 24 in. It then travels to about 21.2 (backward) and then to about 26.8 in. forward due to the pulling force and stretch of the spring. The overall distance that the kicking rod travels is about 5.6 in., which seems to be sufficient to produce enough momentum to kick the ball.

Figure 8-29 shows that the velocity of the kicking rod reaches about 27 in/sec when the rod is pushed near the foremost position. This velocity will produce a momentum of about 310 lbf-sec at 0.5 seconds, as shown in Figure 8-30. To create a momentum graph you may simply right click kicking_rod-l from the browser and choose Plot > Translational Momentum > Z Component. Note that the velocity and momentum are proportional, with the mass of the kicking rod as the scaling factor.

The maximum force required to operate the device is determined to be 5 lb, (maximum force value entered), assuming no friction at any joints. Even though the force data we entered show a polyline in Figure 8-26, the actual force employed for simulation in COSMOSMotion is a spline curve generated.
using the data we entered. What does the force spline look like? We can graph a reaction force for the force ForceAO by right clicking the ForceAO node, and choosing Plot > Reaction Force > Z Component, as shown in Figure 8-31. The force will appear like that of Figure 8-32. Note that this spline curve is smooth and pass through all data points entered.

We turn on friction at both the revolute and the translational joints. Friction for the CamMateTangent joint is currently unavailable in COSMOSMotion.

Delete the simulation result from the browser. Expand the Constraints and then the Joints branch. Right click Revolute and choose Properties. In the Edit Mate-Defined Joint dialog box (Figure 8-33), choose the Friction tab, click the Use Friction, and choose Aluminum Greasy for both Material 1 and Material 2. The Coefficient (mu) will show 0.03. Enter Joint dimensions, Radius: 0.26 and Length: 0.31. Click Apply button to accept the definition. Note that the dimensions entered are the diameter of the pivot pin and the thickness of the handle bar where the joint is located.
Similarly, turn on the friction for the translational joint. Enter 7.25, 7, and 7, for Length, Width, and Height, respectively, as shown in Figure 8-34. Note that the length dimension entered is the sum of both the lower brackets; i.e., 7 and 0.25 in. for the first and second lower brackets, respectively. The width and height of the inner square of the brackets is 7 and 7 in., respectively.

Run a simulation and check the angle of the handle bar.

Even though the friction coefficient is small ($\mu = 0.03$) for both joints, the handle bar hardly moves. Therefore, the force magnitude must be increased. We will follow in general the overall force pattern, and increase the force magnitude to push the handle bar forward.

Delete the simulation result. Expand the Forces and then the Action Only nodes from the browser. Right click the ForceAO node and choose Properties. In the Edit Action-Only Force dialog box, choose the Function tab (see Figure 8-25), and enter the followings:

<table>
<thead>
<tr>
<th>sec</th>
<th>pound_force</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0.1</td>
<td>-8</td>
</tr>
<tr>
<td>0.2</td>
<td>-12</td>
</tr>
<tr>
<td>0.3</td>
<td>-12</td>
</tr>
<tr>
<td>0.4</td>
<td>-8</td>
</tr>
<tr>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>3</td>
</tr>
<tr>
<td>0.7</td>
<td>5</td>
</tr>
<tr>
<td>0.8</td>
<td>3</td>
</tr>
<tr>
<td>0.9</td>
<td>1</td>
</tr>
<tr>
<td>1.0</td>
<td>0</td>
</tr>
<tr>
<td>2.0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 8-35 Operating Force: Small Friction
Click the graph button, the function will appear like the one in Figure 8-35. The maximum force is now 12 lbf. Note that the first half of the force (negative part) is increased more than twice, and the positive portion remains the same. The positive part of the force is kept the same since the spring will contribute partially to the pulling force. The overall force pattern is similar to that of Figure 8-26. The force data entered are results of a few trials-and-errors.

Rerun a simulation. The angle and momentum graphs appear as in Figures 8-36 and 8-37, respectively. The angle graph (Figure 8-36) shows that the handle bar starts at an orientation angle of -7 degrees, and swings forward to about a 12-degree angle, which is acceptable. The handle bar then moves backward to about 10 degrees, and then rests at that configuration due to friction.

The momentum graph (Figure 8-37) shows that when the rod is pushed near the foremost position the momentum of the kicking rod is about 230 lb·sec, which is less than the previous non-friction case. After reviewing the graphs, the 72-lbf force seems to be acceptable. However, this small operating force, 12 lbf, is due to a very small friction force; i.e., friction coefficient \( \mu = 0.03 \). This result also indicates that the spring constant 30 lbf/in seems to be adequate. Next, we will increase the friction coefficient by changing the material from Aluminum Greasy to Aluminum Dry.

Delete the simulation result. Expand the Constraints and then the Joints branch. Right click Revolute and choose Properties. In the Edit Mate-Defined Joint dialog box, choose the Friction tab, and choose Aluminum Dry for both Material 1 and Material 2. The Coefficient (\( \mu \)) will show 0.20. Repeat the same for the translational joint. Run a simulation. The handle bar is hardly moved. That is, the force will have to be increased again.

Delete the simulation results. Right click the ForceAO node to enter the force data. Note that after several attempts, a force that is large enough to move the handle bar, therefore the kicking rod, is about 65 lbf. More specifically, the force data entered are:

![Graphs showing X-rotation angle and Z-momentum](image-url)

Figure 8-36 X-Rotation Angle of the Handle Bar: Small Friction

Figure 8-37 Z-Momentum of the Kicking Rod: Small Friction
Click the graph button, the function will appear like the one in Figure 8-38. Even though the maximum pushing force is now 65 lb, the pulling force (positive portion) remains the same. Are running a simulation, the angle and momentum graphs appear as in Figures 8-39 and 8-40, respectively. Both seem to be reasonable. However, the large operating force, 65 lb, raises a flag. The mechanism requires too large a force for a child to operate.

Note that the simulation engine, *ADAMS/Solver*, is very sensitive to the force data entered. You may encounter problems while carrying out some of the simulations in Task Three. When this happens, simply change the maximum force data, e.g., 65, to a slightly different value, e.g., 63, until a simulation can be completed.

Save your model.
8.4 Result Discussion

Apparently, the biggest concern raised in the simulations is the large operating force. The friction coefficient $\mu = 0.20$, provided by COSMOSMotion for Aluminum-Aluminum contact without lubrication, seems to be physically reasonable. Although this friction coefficient represents an extreme case since in reality some lubricant would be added to the joints to reduce friction resistance. Nevertheless, the force is still too large. As revealed in simulations; i.e., Task Three, the maximum operating force increases from 5 lbf for non-friction, 12 lbf for small friction ($\mu = 0.03$) to 65 lbf for large friction ($\mu = 0.20$). Reducing the friction force at joints, especially the translational joint between the kicking rod and the two lower brackets on the plate, is critical for a successful device. You can easily confirm that the translational joint contributes significantly to the friction encountered in the mechanism by conducting separate simulations where friction is only present in one of the two joints.

The flag raised by the simulation has been observed in the physical device, as shown in Figure 8-41, built by students following the design created in SolidWorks and COSMOSMotion. The physical device confirms that the contact between the kicking rod and the two brackets produces a large friction force, resulting in a large operating force to operate the device. For children with limited physical strength, such a device is unattractive.

In order to reduce the friction, four bearings are added to the device, as shown in Figure 8-42. Two are added to the top surface of the kicking rod, and two are underneath the kicking rod. With the bearings, the friction is significantly reduced. Therefore, a smaller force is required to operate the device. The actual operating force is less than 20 lbf.

8.5 Comments on COSMOSMotion Capabilities and Limitations

Using COSMOSMotion does answer critical questions and help the design process, as demonstrated in this example. However, from this example, a number of limitations in COSMOSMotion have also been encountered. Knowing these limitations will help you use COSMOSMotion more effectively.

First, the simulation engine, ADAMS/Solver, which solves the equations of motion for the mechanism, is not stable. Sometimes when you rerun the same simulation, you could see slightly different results. This problem is more vivid when we ran the same simulation using different computers.
Moreover, the simulation engine produced results, such as the velocity or momentum (see Figures 8-37 and 8-40), with spikes, which is not quite realistic. Some of the spikes disappear or become smaller when we rerun the same simulation.

Second, friction is not supported for a CamMateTangent joint. Therefore, an attempt was made to add a 3D contact joint between the middle pin and the slot of the handle bar. A number of restitution coefficients were used, including 0.5, 0.75, etc. Yet, adding a 3D contact joint significantly slow down the motion of the handle bar and the kicking rod since lots of contact are encountered between the outer surface of the middle pin and the inner surface of the slot when the mechanism is in motion. More contact causes more energy dissipation. However, the slow-down is way too much to grant a physically reasonable simulation. In addition, more spikes appear in graphs, such as in velocity. However, the biggest issue with adding the 3D contact is that the simulation engine becomes extremely sensitive to the initial orientation angle of the handle bar and the magnitude of the operating force. For some simulations, the solution engine encountered problems, for example, force imbalance at certain time steps during the simulation, and terminated the simulation prematurely.

Overall COSMOSMotion is an excellent tool with lots of nice features and capabilities for support of mechanism design and analysis. However, as mentioned in this book numerous times, no software is foolproof. Before creating a simulation model, you always want to formulate your design questions and set up your simulation model and scenarios gearing toward answering these specific questions. You will have to exam the simulation results very carefully and challenge yourself about the validity of the results since if you don’t somebody else will do, usually in a less friendly way.
APPENDIX A: DEFINING JOINTS

Degrees of Freedom

Understanding degrees of freedom is critical in creating successful motion model. The free degrees of freedom of the mechanism represent the number of independent parameters required to specify the position, velocity, and acceleration of each rigid body in the system for any given time. A completely unconstrained body in space has six degrees of freedom, three translational and three rotational. If you add a joint; e.g., a revolute joint to the body, you restrict its movement to rotation about an axis, and the free degrees of freedom of the body are reduced from six to one.

For a given motion model, you can determine its number of degrees of freedom using the Gruebler's count. The mechanism's Gruebler count is calculated using the mechanism's total number of bodies. As mentioned above, each movable body introduces six degrees of freedom. Joints added to the mechanism constrain the system, or remove dofs. Motion inputs, i.e., motion drivers, remove additional dofs.

*COSMOSMotion* uses the following equation to calculate the Gruebler's count:

\[ D = 6M - N - O \]  
\[ (A.1) \]

where \( D \) is the Gruebler count representing the total free degrees of freedom of the mechanism, \( M \) is the number of bodies excluding the ground body, \( TV \) is the number of dofs restricted by all joints, and \( O \) is the number of the motion inputs in the system.

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

If the Gruebler's count is less than zero, the solver will automatically remove redundancies, if possible. For example, you may apply this formula to a door model that is supported by two hinges modeled as revolute joints. Since a revolute joint removes five dofs, the Gruebler's count becomes:

\[ D = (6 \times 1) - (2 \times 5) = -4. \]

The calculated degrees of freedom result is -4, which include five redundant dofs.

Redundancy

Redundancies are excessive dofs. When a joint constrains the model in exactly the same way as another joint (like the door example), the model contains excessive dofs, also known as redundancies. A joint becomes excessive when it does not introduce any further restriction on a body's motion.

It is important that you eliminate redundancies from your motion model while carrying out dynamic analyses. If you do not remove redundancies, you may not get accurate values when you check joint reactions or load reactions.

For example, if you model a door using two revolute joints for the hinges, the second revolute joint does not contribute to constraining the door's motion. *COSMOSMotion* detects the redundancies and ignores one of the revolute joints in its analysis. The outcome may be incorrect in reaction results, yet the
motion is correct. For complete and accurate reaction forces, it is critical that you eliminate redundancies from your mechanism.

For a kinematic simulation where you are interested in displacement, velocity, and acceleration, redundancies in your model do not alter the performance of the mechanism.

You can eliminate or reduce the redundancies in your model by carefully choosing joints. These joints must be able to restrict the same dofs, but not duplicate each other, introducing redundancies. After you decide which joints you want to use, you can use the Gruebler’s count to calculate the dofs and check redundancies.

You may also ask COSMOSMotion to calculate the Gruebler’s count for you. You can simply click the Show simulation control button on top of the graphics screen to bring up the dialog box, as shown in Figure A-1. Note that the DOF field in the dialog box will show the Gruebler’s count if a simulation has been completed. If not, You may click the Calculate button to ask COSMOSMotion to calculate the actual dof. In the message window appearing next (Figure A-2), COSMOSMotion identifies redundant dofs and recalculates the dof for the simulation model.

Joint Types in COSMOSMotion

Before you select a joint to add to your model, you should know what movement you want to restrain for the body and what movement you want to allow. The following table describes the commonly employed joint types in COSMOSMotion and the degrees of freedom they remove.

<table>
<thead>
<tr>
<th>Joint Type</th>
<th>DOF</th>
<th>Removed</th>
<th>Total</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revolute</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>Rotates about an axis</td>
</tr>
<tr>
<td>Translational</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>Translates along an axis</td>
</tr>
<tr>
<td>Cylindrical</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>Translates along and rotates about an axis</td>
</tr>
<tr>
<td>Spherical</td>
<td>3</td>
<td>0</td>
<td>3</td>
<td>Rotates in any direction</td>
</tr>
<tr>
<td>Universal</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>Rotates about two axes</td>
</tr>
<tr>
<td>Screw</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>Coupled rotation and translation along one axis</td>
</tr>
<tr>
<td>Planar</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>Bodies connected by a planar joint move in a plane with respect to each other. Rotation is about an axis perpendicular to the plane</td>
</tr>
<tr>
<td>Fixed</td>
<td>3</td>
<td>3</td>
<td>6</td>
<td>Glues two parts together Parts constrained by a rigid connection constitute a single body</td>
</tr>
</tbody>
</table>
The following provides more details about the joints listed in the table above.

**Revolute Joint**

A revolute joint, as depicted in Figure A-3, allows the rotation of one rigid body with respect to another rigid body about a common axis. The origin of the revolute joint can be located anywhere along the axis about which the bodies can rotate with respect to each other. The joint origin is assigned by COSMOSMotion when you enter COSMOSMotion from SolidWorks.

Orientation of the revolute joint defines the direction of the axis about which the bodies can rotate with respect to each other. The rotational axis of the revolute joint is parallel to the orientation vector and passes through the origin.

![Figure A-3 Revolute Joint Symbol](image)

**Translational Joint**

A translational joint allows one rigid body to translate along a vector with respect to a second rigid body, as illustrated in Figure A-4. The rigid bodies may only translate, not rotate, with respect to each other.

The location of the origin of a translational joint with respect to its rigid bodies does not affect the motion of the joint but does affect the reaction loads on the joint. The location of the joint origin determines where the joint symbol is located.

The orientation of the translational joint determines the direction of the axis along which the bodies can slide with respect to each other (axis of translation). The direction of the motion of the translational joint is parallel to the orientation vector and passes through the origin.

![Figure A-4 Translational Joint Symbol](image)

**Cylindrical Joint**

A cylindrical joint allows both relative rotation and relative translation of one body with respect to another body, as shown in Figure A-5. The origin of the cylindrical joint can be located anywhere along the axis about which the bodies rotate or slide with respect to each other.

Orientation of the cylindrical joint defines the direction of the axis about which the bodies rotate or slide along with respect to each other. The rotational/translational axis of the cylindrical joint is parallel to the orientation vector and passes through the origin.
Spherical Joint

A spherical joint allows free rotation about a common point of one body with respect to another body, as depicted in Figure A-6. The origin location of the spherical joint determines the point about which the bodies pivot freely with respect to each other.

Universal Joint

A universal joint allows the rotation of one body to be transferred to the rotation of another body, as shown in Figure A-7. This joint is particularly useful to transfer rotational motion around corners, or to transfer rotational motion between two connected shafts that are permitted to bend at the connection point (such as the drive shaft on an automobile).

The origin location of the universal joint represents the connection point of the two bodies. The two shaft axes identify the center lines of the two bodies connected by the universal joint. Note that COSMOSMotion uses rotational axes parallel to the rotational axes you identify but passing through the origin of the universal joint.

Screw Joint

A screw joint removes one degree of freedom. It constrains one body to rotate as it translates with respect to another body, as shown in Figure A-8.

When defining a screw joint, you can define the pitch. The pitch is the amount of translational displacement of the two bodies for each full rotation of the first body. The displacement of the first body
relative to the second body is a function of the body’s rotation about the axis of rotation. For every full rotation, the displacement of the first body along the translation axis with respect to the second body is equal to the value of the pitch.

Very often, the screw joint is used with a cylindrical joint. The cylindrical joint removes two translational and two rotational degrees of freedom. The screw joint removes one more degree of freedom by constraining the translational motion to be proportional to the rotational motion.

**Planar Joint**

A planar joint allows a plane on one body to slide and rotate in the plane of another body, as shown in Figure A-9.

The orientation vector of the planar joint is perpendicular to the joint’s plane of motion. The rotational axis of the planar joint, which is normal to the joint’s plane of motion, is parallel to the orientation vector.

![Figure A-9 Planar Joint Symbol](attachment:planar_joint.png)  ![Figure A-10 Fixed Joint Symbol](attachment:fixed_joint.png)

**Fixed Joint**

A fixed joint locks two bodies together so they cannot move with respect to each other. The fixed joint symbol is shown in Figure A-10.
Mapped *SolidWorks* Mates

After you bring an assembly from *SolidWorks* to *COSMOSMotion* and assign components to ground part and moving parts, *COSMOSMotion* automatically maps assembly mates to joints. In most cases, these joints work well for the motion simulation. A selected set of the mapped *SolidWorks* mates, which is commonly employed in assembly, is given in the next table for your reference.

<table>
<thead>
<tr>
<th>Mate Type</th>
<th>Feature 1</th>
<th>Feature 2</th>
<th><em>COSMOSMotion</em> Joint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coincident</td>
<td>Cone</td>
<td>Circular/Arc Edge</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Coincident</td>
<td>Cone</td>
<td>Cone</td>
<td>Revolute</td>
</tr>
<tr>
<td>Coincident</td>
<td>Cylinder</td>
<td>Circular/Arc Edge</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Coincident</td>
<td>Line</td>
<td>Line</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Coincident</td>
<td>Plane</td>
<td>Plane</td>
<td>Planar</td>
</tr>
<tr>
<td>Coincident</td>
<td>Point</td>
<td>Point</td>
<td>Spherical</td>
</tr>
<tr>
<td>Coincident</td>
<td>Point</td>
<td>Sphere</td>
<td>Spherical</td>
</tr>
<tr>
<td>Coincident</td>
<td>Circular/Arc Edge</td>
<td>Circular/Arc Edge</td>
<td>Revolute</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cone</td>
<td>Circular/Arc Edge</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cone</td>
<td>Cone</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cone</td>
<td>Cylinder</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cone</td>
<td>Line</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cylinder</td>
<td>Cylinder</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cylinder</td>
<td>Line</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Cylinder</td>
<td>Circular/Arc Edge</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Line</td>
<td>Circular/Arc Edge</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Point</td>
<td>Sphere</td>
<td>Spherical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Sphere</td>
<td>Sphere</td>
<td>Spherical</td>
</tr>
<tr>
<td>Concentric</td>
<td>Circular/Arc Edge</td>
<td>Circular/Arc Edge</td>
<td>Cylindrical</td>
</tr>
<tr>
<td>Distance</td>
<td>Cone</td>
<td>Cone</td>
<td>Revolute</td>
</tr>
<tr>
<td>Distance</td>
<td>Plane</td>
<td>Plane</td>
<td>Planar</td>
</tr>
</tbody>
</table>

Note that a complete list of mapped mates can be found in *COSMOSMotion* help. *COSMOSMotion* help system can be viewed by choosing from the pull-down menu, *Help > COSMOS > Help on COSMOS Education Edition.*
APPENDIX B: THE UNITS SYSTEM

The mass unit lbₘ is not quite common to many engineers. The basic physical quantities involved in determining a units system are length, time, mass, and force. These four basic quantities are related through Newton's second law,

\[ F = ma \]  \hspace{1cm} (B.1)

where \( F \), \( m \), and \( a \) are force, mass, and acceleration (length per second square), respectively.

When the unit lbₘ for mass is employed, the force unit will be determined by length (in.), mass (lbₘ), and second (sec) through Eq. B.1; i.e.,

\[ 1 \text{ lbₘ in/sec}^2 \text{ (force)} = 1 \text{ lbₘ (mass)} \times 1 \text{ in/sec}^2 \text{ (acceleration)} \]  \hspace{1cm} (B.2)

where the force unit, lbₘ in/sec², is a derived unit.

From Eq. B.2, a 1 lbₘ in/sec² force will generate a 1 in/sec² acceleration when applied to a 1 lbₘ mass block, as shown in Figure B-1a. The same block will weigh 1 lb₇ on earth (see Figure B-1b), where the gravitational acceleration is assumed 386 in/sec²; i.e.

\[ 1 \text{ lb₇ (force)} = 1 \text{ lbₘ (mass)} \times 386 \text{ in/sec}^2 \text{ (acceleration)} \]  \hspace{1cm} (B.3)

\[ \begin{align*}
F &= 1 \text{ lbₘ in/sec}^2 = 1 \text{ lbₘ} \times 1 \text{ in/sec}^2 \\
W &= 1 \text{ lb₇} = 1 \text{ lbₘ} \times 386 \text{ in/sec}^2 \\
F &= 1 \text{ lb₇ = 1 lbₘ \times 386 in/sec}^2 \\
W &= 1 \text{ lb₇} = 1 \text{ lbₘ} \times 386 \text{ in/sec}^2 \\
F &= 1 \text{ lb₇ = 1 lb₇ s}^2/\text{in} \times 1 \text{ in/sec}^2 \\
W &= 1 \text{ lb₇ s}^2/\text{in} \times 386 \text{ in/sec}^2 = 386 \text{ lb₇}
\end{align*} \]

(a) A 1 lbₘ in/sec² Force Applied to a 1 lbₘ Mass Block

(b) A 1 lb₇ Force Applied to a

(c) A 1 lb₇ Force Applied to a

1 lbₘ Mass Block

1 lb₈ sec²/in Mass Block

Figure B-1 Forces Applied on Blocks of Different Masses
Therefore, from Eqs. B.2 and B.3, we have \( 7 \, \text{lb}_f = 386 \, \text{lb}_m \, \text{in/sec}^2 \), and the force in \( \text{lb}_m \, \text{in/sec}^2 \) unit is 386 times smaller than that of \( \text{lb}_f \) that we are more used to. When you apply a 7 lb force to the same mass block, it will accelerate 386 in/sec^2, as shown in Figure B-1b.

On the other hand, we have the mass unit, \( 7 \, \text{lb}_m = \frac{1}{386} \, \text{lb}_f \, \text{sec}^2 / \text{in} \). It means that a \( \frac{1}{386} \, \text{lb}_f \) mass block is 386 times smaller than that of a 7 lb, sec^2/in block. Therefore, a \( \frac{1}{386} \, \text{lb}_f \) sec^2/in block will weigh 386 lb, on earth. When applying a \( \frac{1}{386} \, \text{lb}_f \) force to the mass block, it will accelerate at a 7 in/sec^2 rate, as illustrated in Figure B-1c.

The mass unit slug that we are more familiar with is defined as

\[
1 \, \text{lb}_f (\text{force}) = 1 \, \text{slug (mass)} \times 1 \, \text{ft/sec}^2 \, (\text{acceleration}) = 386 \, \text{lb}_m \, \text{in/sec}^2 = 32.2 \, \text{lb}_m \, \text{ft/sec}^2
\]  

(B.4)

Therefore, we have 1 slug = 32.2 lb_m.
APPENDIX C: IMPORTING Pro/ENGINEER PARTS AND ASSEMBLIES

From time to time when you use COSMOSMotion for simulations, you may encounter the need for importing solid models from other CAD software, such as Pro/ENGINEER. SolidWorks provides an excellent capability that support importing solid models from a broad range of software and formats, including Parasolid, ACIS, IGES (Initial Graphics Exchange Standards), STEP (STandard for Exchange of Product data), SolidEdge, Pro/ENGINEER, etc. For a complete list of supported software and formats in SolidWorks, please refer to Figure C-1. You may access this list by choosing File > Open from the pull-down menu, and pull down the Files of type in the File Open dialog box.

In this appendix, we will focus on importing Pro/ENGINEER parts and assemblies. Hopefully, methods and principles you learn from this appendix will be applicable to importing solid models from other software and formats.

SolidWorks provides capabilities for importing both part and assembly. Users can choose two options in importing solid model. They are Option 1: importing solid features and Option 2: importing just geometry. Importing solid features may bring you a parametric solid model that just like a SolidWorks part that you will be able to modify. On the other hand, if you choose to import geometry only, you will end up with an imported feature that you cannot change since all solid features are lumped into a single imported geometry without any solid features nor dimensions.

Importing geometry is relatively straightforward. In general, SolidWorks does a good job in bringing in Pro/ENGINEER part as a single imported geometry. In fact, several other translators, such as IGES and STEP, support such geometric translations well. IGES and STEP are especially useful when there is no direct translation from one CAD to another.

Importing solid models with solid features is a lot more challenging, in which solid features embedded in the part geometry, such as holes, chamfers, etc., must be identified first. In addition, sketches that were employed for generating the solid features must be recovered and the feature types, for instance revolve, extrude, sweep, etc., must be identified. With virtually infinite number of possibilities in creating solid features, it is almost certain that you will encounter problems while importing solid models with feature conversion. Therefore, if you do not anticipate making design changes in SolidWorks, it is highly recommend that you import parts as a single geometric feature.

We will discuss the approaches of importing parts, and then importing assemblies. In each case, we will try both options; i.e., importing solid features vs. importing geometry. We will use the gear train example employed in Lesson 6 as the test case and as an example for illustrations.
The Gear Train Example in Pro/ENGINEER

The gear train assembly consists of one part and three subassemblies. If you have access to Pro/ENGINEER, you may want to open the final assembly, gear_train_final.asm, to check the assembled gear train shown in Figure C-2. There are four components in this assembly: gbox_housing.prt, gbox_input.asm, gbox_middle.asm, and gbox_output.asm. The input and output gear assemblies consist of one gear each, Pinion 1 and Gear 2, respectively. The middle gear assembly has two gears, Gear 1 and Pinion 2. The four spur gears form two gear pairs: Pinion 1 and Gear 1, and Pinion 2 and Gear 2, as illustrated in Figure C-2. Gear 1 and Pinion 2 are mounted on the same shaft.

There are 22 distinct parts in this assembly, as listed in Table C-1. All the parts and assemblies can be found in folder Appendix C.

Table C-1 List of Parts and Assemblies in Appendix C Folder

<table>
<thead>
<tr>
<th>Part/Subassemblies</th>
<th>Part Names</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>gbox_housing.prt</td>
<td>wheel_gbox_shaft_input.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_pinion_1s.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>spacer 12×18×5mm.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>spacer 12×20×1mm.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bearing 12×18×8mm.prt (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>spacer 10×18×014mm.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_sft_mid_washer.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>screw_tapper_head 5×15.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>screw_set_tip 6×6.prt (2)</td>
<td></td>
</tr>
<tr>
<td>gbox_input.asm</td>
<td>wheel_gbox_pinion_2s.prt</td>
<td>Pinion 2</td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_gear_1s.prt</td>
<td>Gear 1</td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_shaft_mid_pinion.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_shaft_mid_gear.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bearing 12×18×8mm.prt (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>screw_tapper_head 5×28.prt (6)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_sft_mid_washer.prt (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>screw_tapper_head 5×15.prt (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>align_pin 4×27mm.prt (2)</td>
<td></td>
</tr>
<tr>
<td>gbox_middle.asm</td>
<td>wheel_gbox_gear_2s.prt</td>
<td>Gear 2</td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_connect_wheel.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bear_tap_roller25×47×15mm.prt</td>
<td></td>
</tr>
<tr>
<td></td>
<td>screw_straight_head 4×15.prt (10)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>align_pin 4×20mm.prt (2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>wheel_gbox_connect_wh_setscrew.prt</td>
<td></td>
</tr>
</tbody>
</table>
sketches in the graphics screen by clicking their names listed in the browser. In addition, the back plate (Extrudel in the browser) is recognized incorrectly. Certainly, SolidWorks is capable of importing some parts correctly and completely, especially, when the solid features are relatively simple (but not this gear housing part).

If you take a closer look at any of the successful solid features, for example ExtrudeS, you will see that the sketches (for example Sketch3 of ExtrudeS) of the solid features do not have complete dimensions. Usually a (—) symbol is placed in front of the sketch, indicating that the sketch is not fully defined.

Apparently, this translation is not satisfactory. Unfortunately, this translation represents a typical scenario you will encounter for the majority of the parts. In many cases, it may take only a small effort to repair or re-create wrong or unrecognized solid features. However, when you translate an assembly with many parts, the effort could be substantial.

Option 2: Importing Geometry

Importing geometry is more straightforward and has a higher successful rate than that of importing solid features.

Repeat the same steps to open the gear housing part, gboxjousingprt. In the Pro/ENGINEER to SolidWorks Converter dialog box (Figure C-8), choose Import geometry directly (default), and then Knitting (default) in order to import solid models instead of just surface models. Note that if you choose BREP (Boundary Representation), only boundary surfaces will be imported. Click OK.
The conversion process will start. After about a minute or two, the converted model will appear in the graphics screen, as shown in Figure C-9. In addition, an entity Imported will appear in the browser (Figure C-9). As mentioned earlier, there will be no parametric solid feature with dimensions and sketch converted if you choose Option 2. However, the geometry converted seems to be accurate. All the geometric features in Pro/ENGINEER shown in Figure C-3 were included in this imported feature. This translation is successful. Since we do not anticipate making any change to the gear housing, this imported part is satisfactory. The gear housing part, gbox_housing, sl1dprt, employed in Lesson 6 was created by using Option 2.

**Importing Pro/ENGINEER Assembly**

We will import the input gear assembly (gbox_input.asm) shown in Figure C-10 using both options. We will try Option 1 first; i.e., importing solid features. As shown in Figure C-10 (Pro/ENGINEER Model Tree window), there are 11 parts (and several datum features) in this assembly. SolidWorks will try to import this assembly as well as the 11 parts from Pro/ENGINEER.

**Option 1: Importing Solid Features**

Repeat the same steps to open the input gear assembly, gbox_input.asm. In the Pro/ENGINEER to SolidWorks Converter dialog box (Figure C-11), choose Use feature import for all parts, and choose Overwrite for If same name SolidWorks file is found (just in case you have SolidWorks files with the same file names in the same folder). Choose Import material properties and Import sketch/curve entities. Click Import. The conversion process will start.

You will see sketches, solid features, and solid models appear in the graphics screen. After about a minute, the translation process is completed. The converted assembly and the browser with parts listed are shown in Figure C-12.

As shown in Figure C-12, parts are not completely converted. Major solid features are missing, for example, the pinion 1 (wheel_gbox_pinion_1s<1>), where most solid features are not converted. If you expand the part, you will see that only one extrude and one cut features were converted, the remaining entities are mostly sketches. There are only three simple parts converted successfully, spacer_12x18x5mm<1>, spacer_12x20x1mm<1>, and spacer_10x18x014mm<1>. In addition, the Mates branch in the browser is empty, implying that no assembly mates have been imported.

Apparently, this translation is not satisfactory. A non-trivial effort will be have to be spent in order to reconstruct the solid features (therefore, solid models) as well as the final assembly.
Importing geometry is also more straightforward for assembly and has a higher rate of success.

Repeat the same steps to open the input gear assembly, gbox_input.asm. In the Pro/ENGINEER to SolidWorks Converter dialog box (Figure C-13), choose Use body import for all parts (default), and then Knitting (default) in order to import solid models. Choose Overwrite for If same name SolidWorks file is found, and choose Import material properties and Import sketch/curve entities. Click Import. The conversion process will begin.

After about a minute or two, the converted assembly will appear in the graphics screen, as shown in Figure C-14. The assembly and all 11 parts seem to be correctly imported. If you expand any of the part branch, for example, the gear (wheel_gbox_pinion_fs<l>), you will see an imported feature listed, as
depicted in Figure C-14. Again, there is no solid feature converted in any of the parts. In addition, the Mates branch is empty.

Since we do not anticipate making any change to this input gear assembly, this imported assembly is satisfactory, except it does not have any assembly mates. Assemble all 11 parts (may be more for some cases) will take a non-trivial effort. Since we do not anticipate making change in how these parts are assembled, we will merge all 11 parts into a single part.

In SolidWorks, you can join two or more parts to create a new part in an assembly. The merge operation removes surfaces that intrude into each other’s space, and merges the parts into a single solid volume. We will insert a new part into the assembly and merge all 11 parts into that new part. Choose from the pull-down menu Insert > Component > New Part. In the Save As dialog box, enter gbox Input for part name, and click Save. SolidWorks is expecting you to select a plane or a flat face to place a sketch for the new part. Click a plane or planar face on a component, for example pick the assembly Front plane from the browser, the Front plane will appear in the graphics screen (Figure C-15). The new part gboxinput will appear in the browser. In the new part, a sketch opens on the selected plane. Close the sketch. Because we are creating a joined part, we do not need a sketch.

Next, we will select all 11 parts and merge them into the new part.

From the browser, click the first part wheel box_shaftinput, press the Shift key, and then click the last part, screw_setjip_6*6. All 11 parts will be selected.

From the pull-down menu, choose Insert > Features > Join. The Join window will appear (overlapping with the browser) as shown in Figure C-16. In the Join window, all 11 parts are listed. All you have to do is to click the checkmark on top to accept the parts. Save the assembly (and the part), and then close the whole assembly.

Now open the part gbox Input. Make sure you open gbox input, sldprt instead of gboxInput. sldasm. The part gbox input will appear in the graphics screen. In addition, all entities belong to this part will be listed in the browser, as shown in Figure C-17. Note that there is an arrow symbol -> to the right of the root entity, gbox Input. This symbol indicates that these entities enclosed in this part refer to other parts or assembly. Note that the Join branch has the same symbol. Expand the Join branch, you will see 11 parts listed, all with arrows, pointing to the actual parts currently in the same folder. When the link is broken; i.e., when the referring parts are removed from the folder, a question mark symbol will be added to the arrow.

The three gear parts, gbox Input, sldprt, gboxmiddle.sldprt, and gbox output, sldprt, employed for Lesson 6 were created following the approach discussed. One axis in each part that passes through the center hole of the gear was created simply by intersecting two planes, for example, Top and Right planes for the axis in gbox Input, sldprt, as shown in Figure C-18. These axes are necessary for creating gear pairs, as discussed in Lesson 6.
Figure C-16

Figure C-17

Figure C-18
Table of Contents

1. Introduction to COSMOSMotion
2. A Ball Throwing Example
3. A Spring Mass System
4. A Simple Pendulum
5. A Slider-Crank Mechanism
6. A Compound Spur Gear Train
7. Cam and Follower
8. Assistive Device for Wheelchair Soccer Games