Working Model™ Tutorial

Information in this document is subject to change without notice and does not represent a commitment on the part of MSC.Software Corporation. The software described in this document is furnished under a license agreement or non-disclosure agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license or non-disclosure agreement. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, for any purpose without the express written permission of MSC.Software Corporation.

© Copyright MSC.Software Corporation 2004. All rights reserved. Published and printed in the U.S.A.

Portions © 1992-1995 Summit Software Company.

MSC and the MSC.Software logo are registered trademarks of MSC.Software Corporation.
Smart Editor, Working Model, Working Model Basic and WM Basic are trademarks of MSC.Software Corporation.
Mac and MacOS are registered trademarks of Apple Computer, Incorporated.
Microsoft, Windows, and Windows NT are registered trademarks of Microsoft Corporation.
PowerPC is a registered trademarks of International Business Machines Corporation.
MATLAB is a registered trademark of the MathWorks, Incorporated.

All other brand or product names are trademarks or registered trademarks of their respective holders.

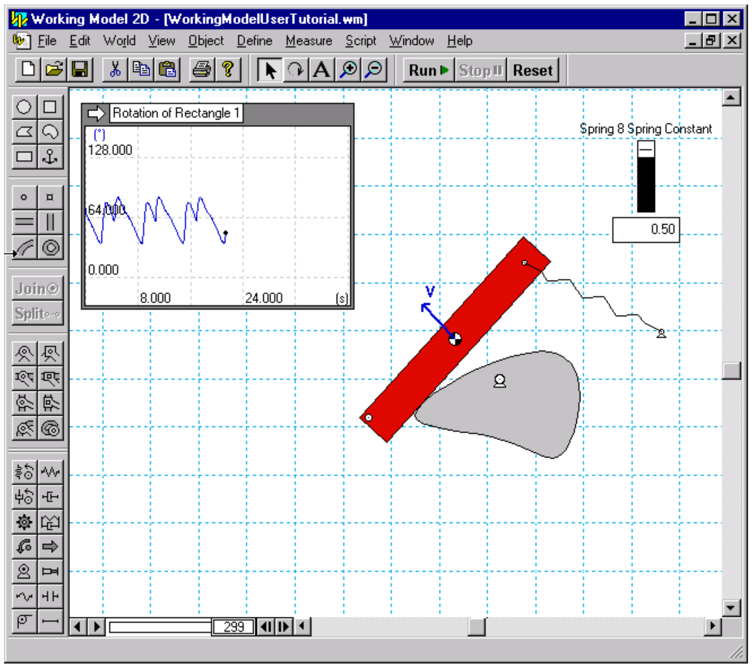
Last Revised 06/01/2004

http://www.workingmodel.com

Contents

EXERCISE 1

Introductory Demonstration


Working Model 2D - [WorkingModelUserTutorial.wm]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Rotation of Rectangle 1
128.000
64.000
0.000
8.000
24.000
(s)
Spring 8 Spring Constant
0.50
Join
Split
299

Concepts for Exercise 1:

- Drawing simple shapes
- Using graphs to measure quantities
- Using simple pin joints, springs, and motors
- Changing air resistance
- Creating a cam-follower mechanism

1.1. Starting Working Model

1. **Ensure that Working Model is installed on your computer.**
2. **From the Start Menu, click on Programs, then Working Model, and then Working Model 2D. This opens a new Working Model document.**

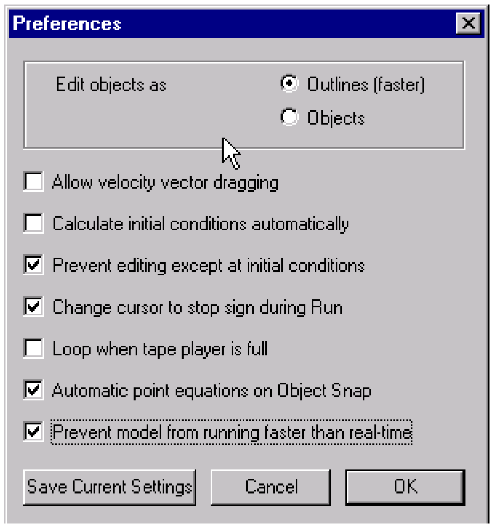
1.2. Setting Up the Workspace

Before starting the exercises, change a few settings:

1. **From the World menu, select Preferences.**

   The Preferences dialog appears.

Figure 1-1
Preferences dialog



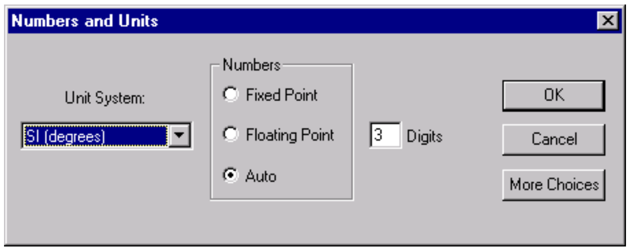
Preferences

Edit objects as [✓ Outlines (faster)] Objects

- [ ] Allow velocity vector dragging
- [ ] Calculate initial conditions automatically
- [x] Prevent editing except at initial conditions
- [x] Change cursor to stop sign during Run
- [ ] Loop when tape player is full
- [x] Automatic point equations on Object Snap
- [x] Prevent model from running faster than real-time

2. **Check the box labeled “Prevent model from running faster than real-time” and click [OK].**
3. **From the View menu, select Numbers and Units.**

   The Numbers and Units dialog appears.


Numbers and Units

Unit System: SI (degrees)

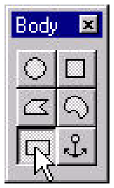
Numbers: Fixed Point / Floating Point / [✓ Auto]

3 Digits

Figure 1-2
Numbers and Units dialog

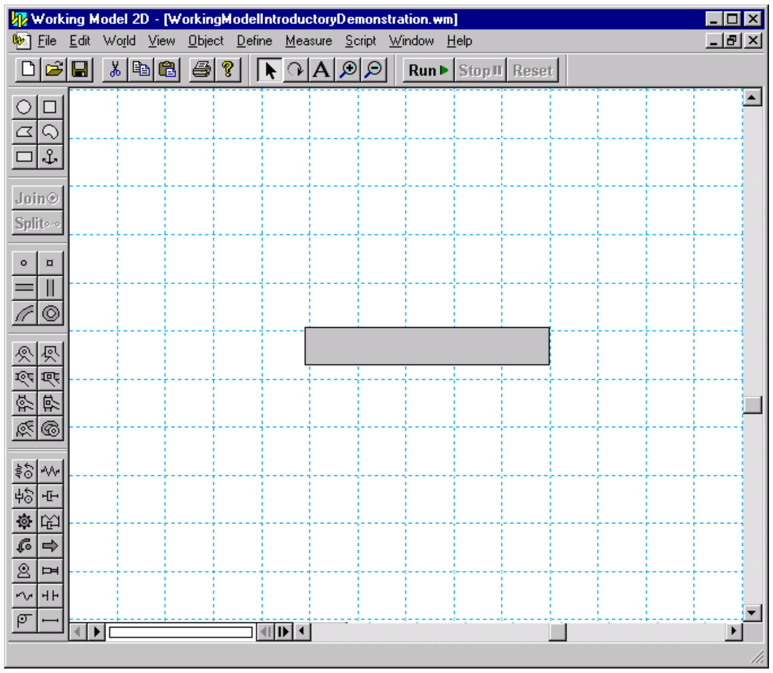
4. **Change the Unit System to “SI (degrees)” and click [OK].**

1.3. Creating a Falling Block

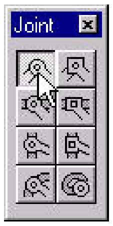


1. **The first simulation is Newton’s first experiment, dropping a block.**
2. **To draw a rectangle, click on the Rectangle tool, then click in the workspace and draw a long thin rectangular block.**
3. **To run the simulation and see the block fall due to gravity, click Run ▶.**
4. **To stop the simulation, click Stop ⏸. Click Reset to reset the simulation.**

Figure 1-3
Rectangular block


Working Model 2D - [WorkingModelIntroductoryDemonstration.wm]
File Edit World View Object Define Measure Script Window Help
Run
Stop
Reset
Join
Split

1.4. Making a Pendulum

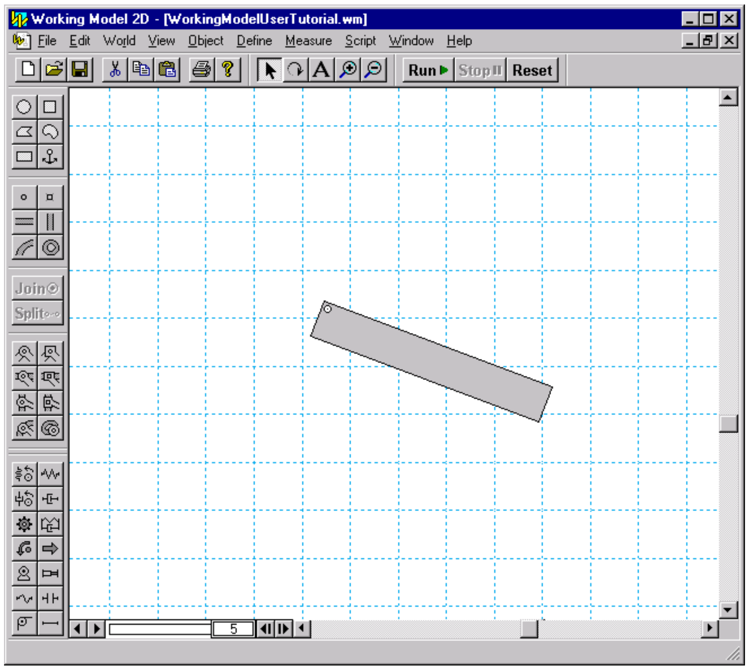


1. **To make a pendulum, click on the Pin joint tool and then click on the upper left-hand corner of the rectangle.**

2. **Click Run ▶ and observe the pendulum's motion.**

3. **Click Stop ⏸ and Reset.**

Figure 1-4
Pendulum


Working Model 2D - [WorkingModelUserTutorial.wm]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
5

1.5. Adding a Velocity Vector

1. **To add a velocity vector, click on the rectangle.**
2. **From the Define menu, click on Vectors and then Velocity.**
3. **Click Run ▶ and observe that the vector changes direction and magnitude as the pendulum moves.**
4. **Click Stop ⏸ and Reset.**

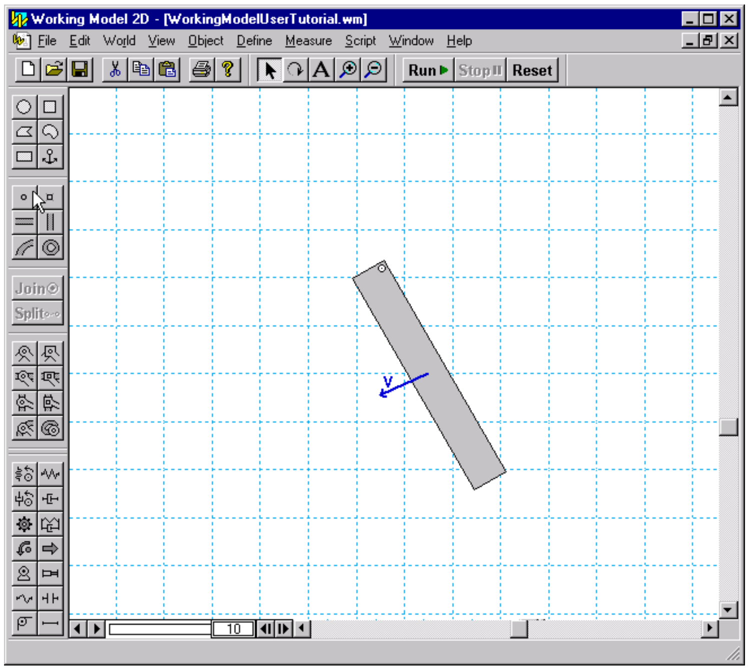
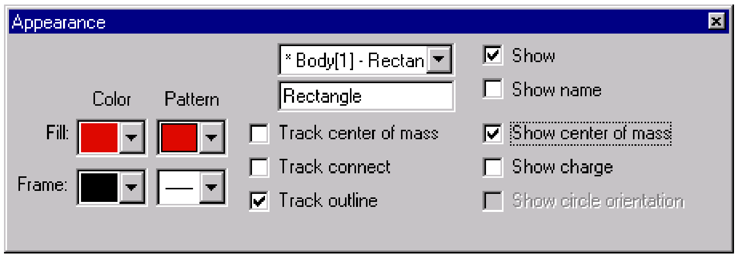

Working Model 2D - [WorkingModelUserTutorial.wm]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Join
Split
10
v

Figure 1-5
Pendulum with velocity vector

1.6. Changing an Object’s Appearance

1. **To change the appearance of the rectangle, double-click on it. Under the Windows menu, select Appearance. Change the fill color and click the box titled "Show center of mass."**


Appearance

* Body[1] - Rectan

Rectangle

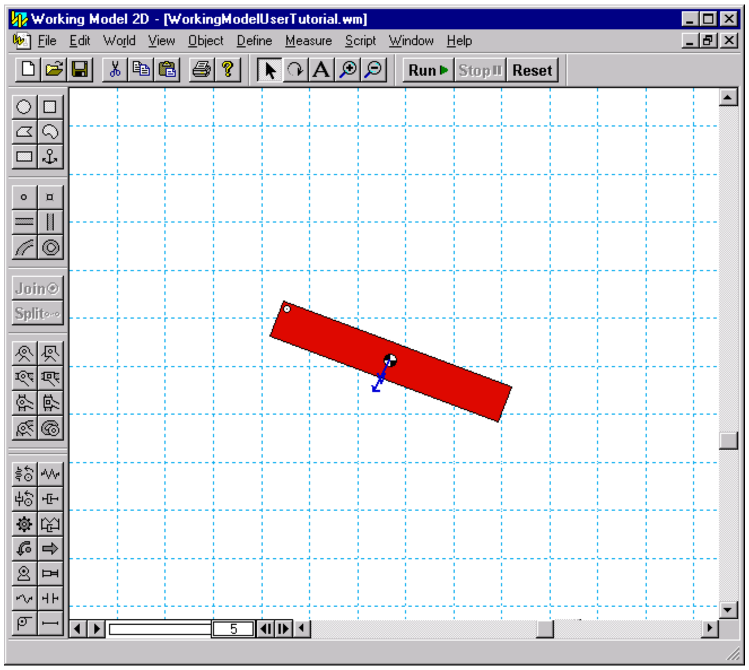
	Color	Pattern
Fill:	🟥	🟥
Frame:	⬛	—

- [ ] Track center of mass
- [ ] Track connect
- [x] Track outline
- [x] Show
- [ ] Show name
- [x] Show center of mass
- [ ] Show charge
- [ ] Show circle orientation

Figure 1-6
Appearance dialog

2. **Close the Appearance window and run the simulation again. Notice that changing the rectangle’s appearance does not affect its motion.**

Figure 1-7
Red rectangular block showing center of mass

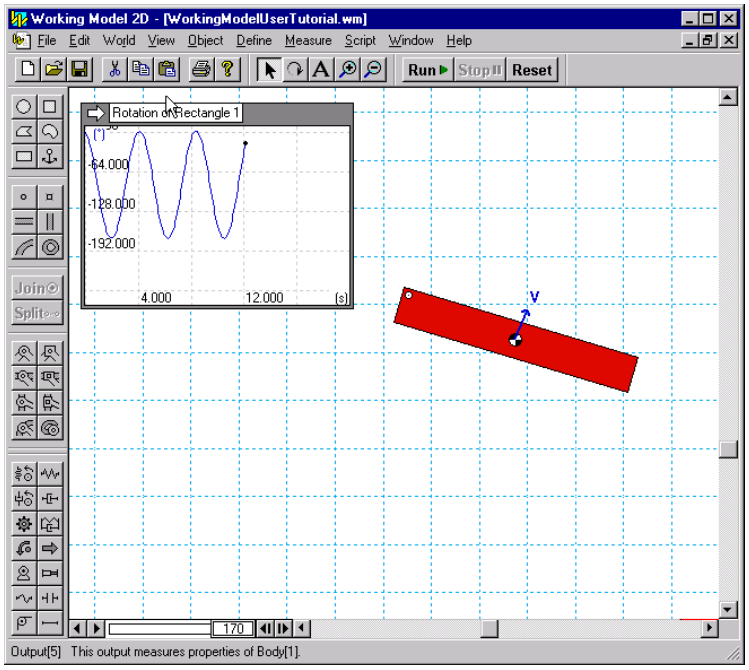

Working Model 2D - [WorkingModelUserTutorial.wm]
File Edit World View Object Define Measure Script Window Help
Run
Stop
Reset
Join
Split
5

1.7. Graphing the Pendulum's Motion

1. **To graph the pendulum's motion, click on the rectangle. Under the Measure, select Position, then select Rotation Graph.**
2. **To collect data, click Run ▶, the data can be displayed as a graph, a bar chart, or a number. (Notice that the data display can be changed while running the simulation.)**
3. **From the graph, the amplitude and frequency of the pendulum's motion can be determined.**
4. **To make the graph larger, click on the graph and drag its lower right-hand corner to the right.**

Figure 1-8

Graphing the pendulum’s motion


Working Model 2D - [WorkingModelUserTutorial.wm]
File Edit World View Object Define Measure Script Window Help
Run Stop Reset
Rotation of Rectangle 1
(°)
-64.000
-128.000
-192.000
4.000
12.000
(s)
v
Join
Split
170

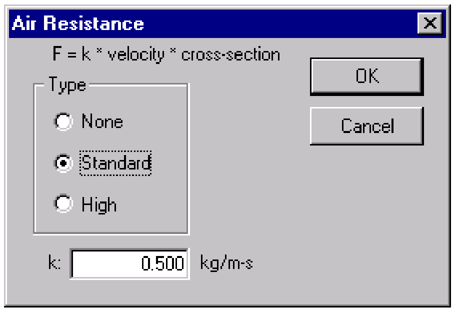
Output[5] This output measures properties of Body[1].

1.8. Adding Air Resistance

1. **Under the World menu, select Air Resistance, click on Standard, and enter a small value, e.g., 0.5 kg/(m * s).**

   Note: Working Model was designed to be easy-to-use. For example, in this exercise, the only time you need to touch the keyboard is to enter the value 0.5.

Figure 1-9
Air resistance dialog


Air Resistance

F = k * velocity * cross-section

Type

None

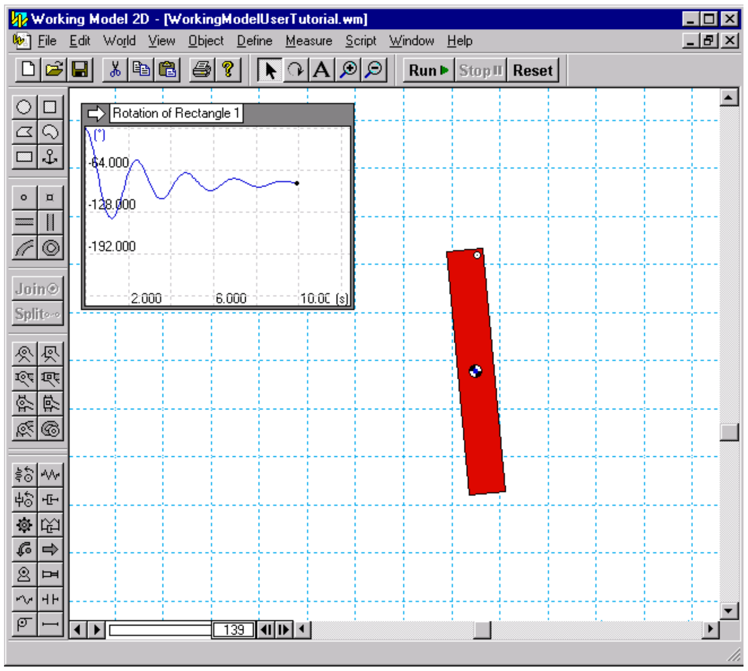
[✓ Standard]

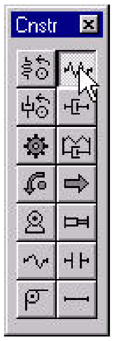
High

k: 0.500 kg/m-s

2. Click **Run** and observe the exponentially decaying oscillations and notice that the pendulum's center of mass comes to rest directly below the pin. Click **Stop** and **Reset**.

Figure 1-10
Pendulum under air resistance



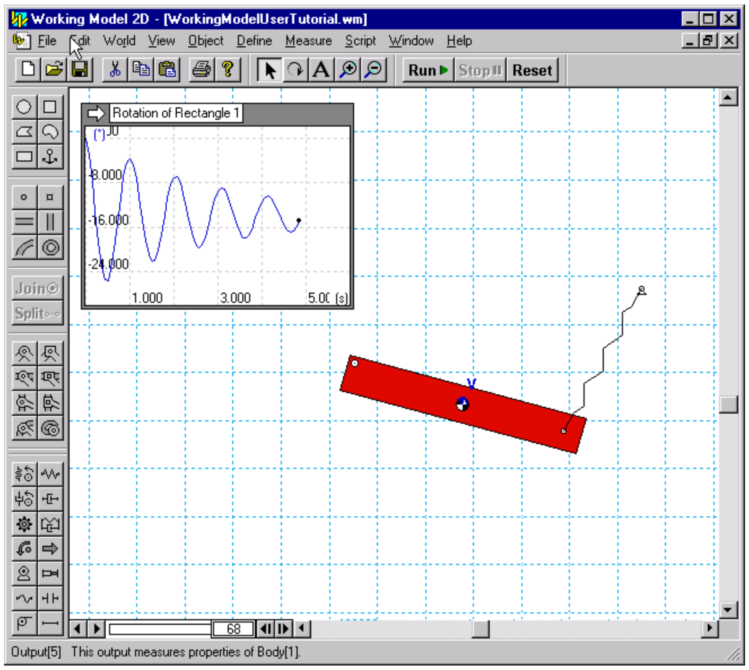


1.9. Adding a Spring

1. **To add a spring, click on the Spring tool. Click on the upper right-hand corner of the block and stretch the spring up and to the left.**

2. **Click Run ▶ and observe the pendulum's higher natural frequency and new equilibrium position. Click Stop ❚❚ and Reset.**

Figure 1-11
Pendulum constrained by a pin joint and a spring



1.10. Controlling the Spring Constant

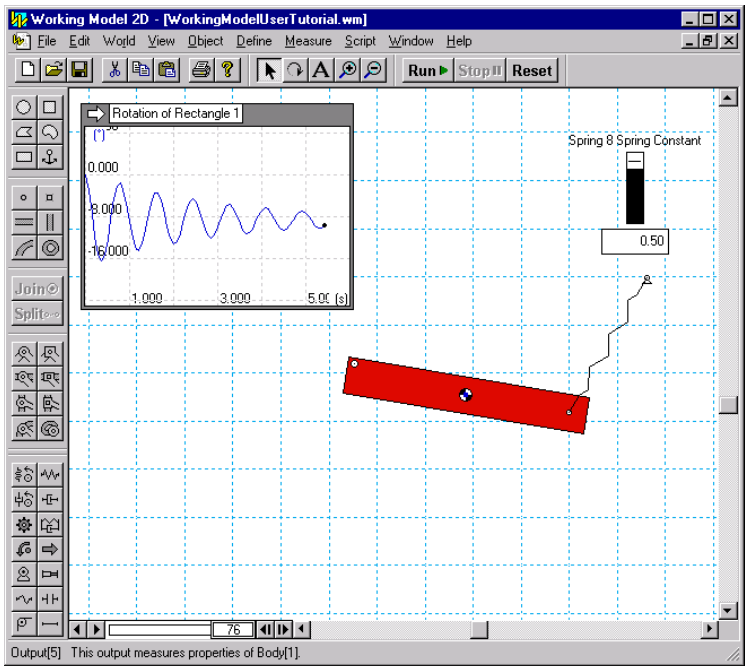
1. **To control the spring constant, select the spring. Under the Define menu, select New Control, then select Spring Constant.**

2. **The slider that controls the spring constant will appear in the upper left-hand corner of the workspace. To change the slider's location to the right-hand corner of the workspace, click on the title and drag it to the new location.**

3. **To see the effect of varying the spring constant, click Run ▶ and observe that the equilibrium angle of the pendulum is a function of the spring-constant (move the slider up and down while the simulation is running).**

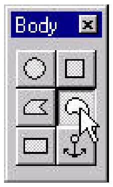
Figure 1-12

Adding a spring constant control



Output[5] This output measures properties of Body[1].

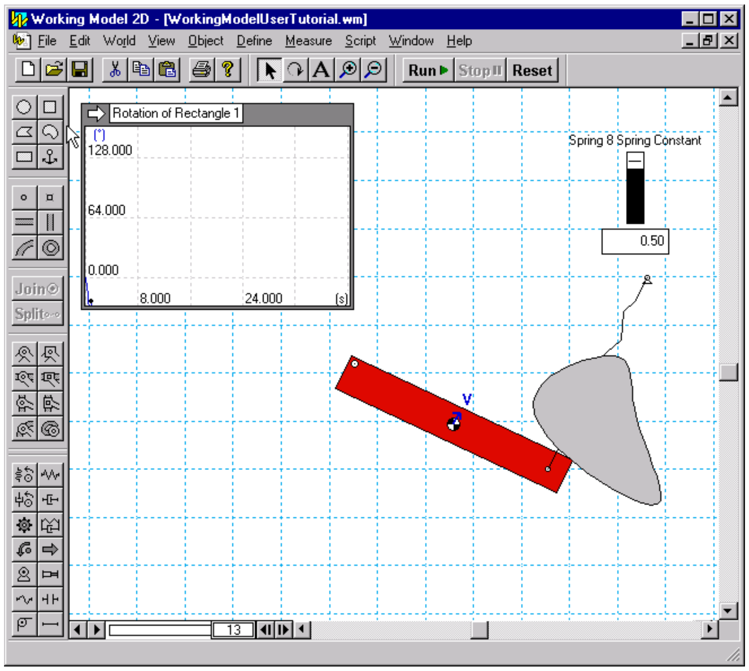
1.11. Collisions with a Smooth Polygon



1. **To create a smooth polygon, click on the Curved Polygon tool, click in the workspace in a few locations. Double-click to close the polygon.**

2. Click **Run ▶** to start the simulation and observe that the curved polygon bounces and rolls on top of the rectangle. Automatic collision and contact is a very useful feature in Working Model, and the elastic and frictional properties of objects may be varied. Click **Stop ⏸** and **Reset**.

Figure 1-13
Collision of cam object with rectangle

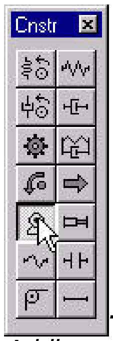


1.12. Smart Editor

1. Working Model's Smart Editor allows the user to change an object's position and orientation while keeping the existing constraints intact.

2. To change the orientation of the rectangle, click on the rectangle and drag the mouse to rotate the rectangle counterclockwise.

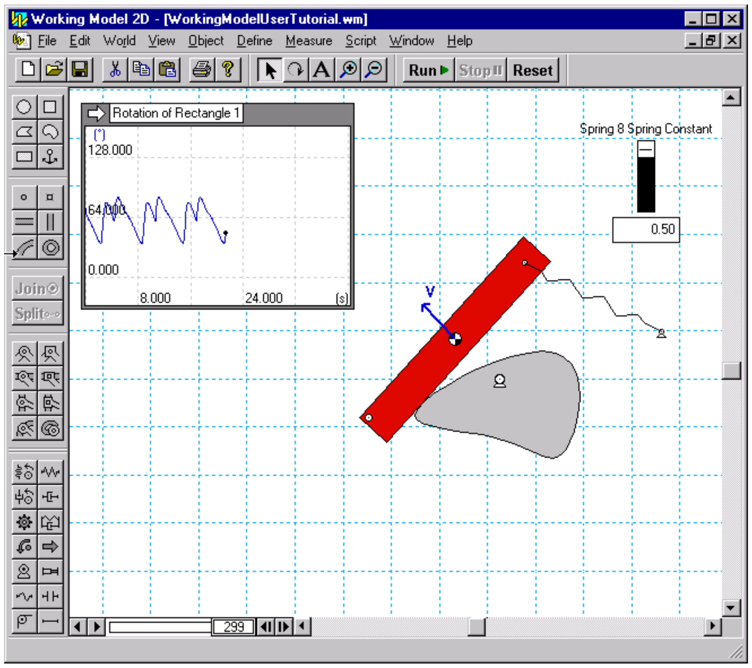
1.13. Creating a Cam-Follower Mechanism



-14
Adding a motor to create a cam- follower mechanism

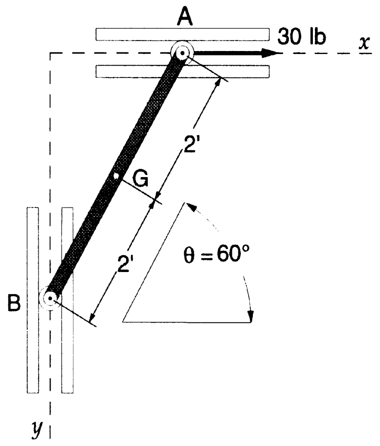
1. **To add a motor to the curved polygon, click on the Motor tool, and then click on the left-hand corner of the curved polygon.**

2. **Click Run and observe that the rectangle's motion is determined by the shape of the curved polygon and the motor's speed.**



E X E R C I S E   2

A Double-Slotted Rod



The slender bar AB weighs 60 lbs. and moves in the vertical plane with its ends constrained to follow smooth horizontal and vertical guides. The bar is initially at rest in a position such that θ = 60°. A 30-lb. force in the positive x-direction is applied at A. Calculate the initial angular acceleration of the bar and the initial forces on the small end rollers at A and B.

Exercise 2 Concepts:

- Utility windows
- Changing the unit system
- Precise placement of points and slots
- Joining points and slots to create slot joints
- Creating and scaling forces
- Displaying and scaling vectors
- Meters

2.1. Introduction

This exercise utilizes three components: a rod, a horizontal slot, and a vertical slot. A rectangular body will model the rod; two slot joints positioned on the x and y axes will model the horizontal and vertical guides. The rectangle will be drawn, sized, and then joined to the two slots. A force will be applied to the rectangle and the resulting angular velocity will be measured.

2.2. Setting Up the Workspace

In this exercise, you will use the English unit system of pounds and feet. To verify the current unit system:

1. **Choose Numbers and Units... from the View menu.**

   The Numbers and Units dialog appears.

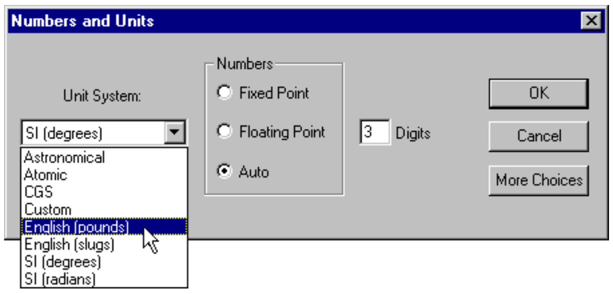


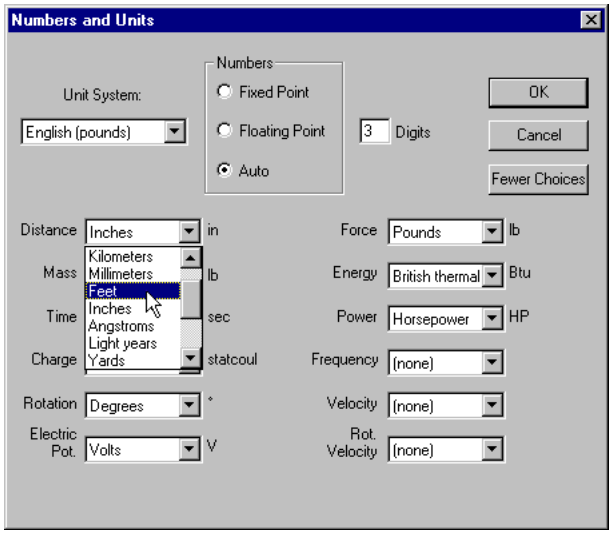
Figure 2-1
Numbers and Units dialog

2. **Choose English (pounds) from the Unit System pop-up menu if it is not already selected (Figure 2-1).**

The default distance unit in the English system is inches. To change this to feet:

3. **Click More Choices.**

   The dialog box expands to allow custom settings for various units.



Unit System: English (pounds)

Numbers: ( ) Fixed Point ( ) Floating Point [✓ Auto]

3 Digits

Distance	Inches	in	Force	Pounds	lb
Mass		lb	Energy	British thermal	Btu
Time		sec	Power	Horsepower	HP
Charge		statcoul	Frequency	(none)	
Rotation	Degrees	°	Velocity	(none)	
Electric Pot.	Volts	V	Rot. Velocity	(none)	

Figure 2-2
Numbers and Units dialog (expanded)

4. Choose Feet from the Distance pop-up menu (Figure 2-2).

5. Click OK.

In addition, you will need the x-y axes for this exercise. If the x-y axes are not currently displayed, do the following:

1. Choose Workspace from the View menu.

On MacOS systems, this leads to a submenu of workspace options which can be set. Choosing Workspace... from this submenu leads to a dialog box which allows you to set multiple options at once. On Windows systems, there is no Workspace submenu; the menu command leads directly to the Workspace dialog.

2. On MacOS systems, choose X,Y Axes from the Workspace submenu (Figure 2-3). On Windows systems, check the box next to X,Y Axes in the Workspace dialog (Figure 2-4).

The simulation window should look similar to Figure 2-5.

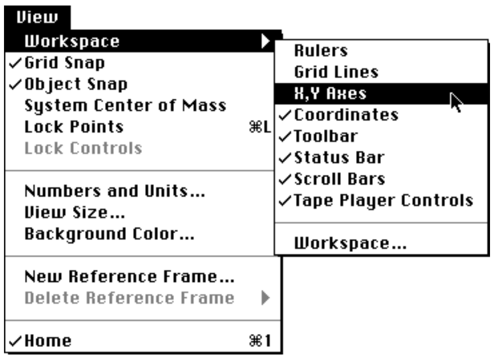
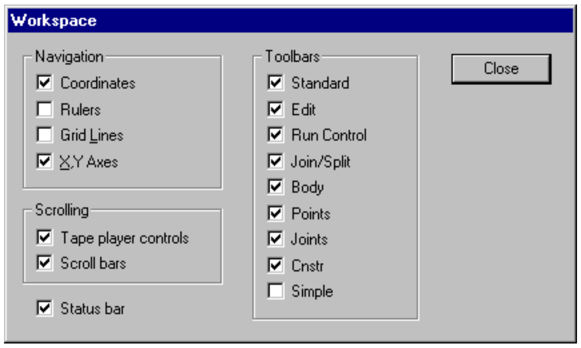


Figure 2-3
Workspace submenu (MacOS only)


Workspace

Navigation
- [✓] Coordinates
- [ ] Rulers
- [ ] Grid Lines
- [✓] X,Y Axes

Scrolling
- [✓] Tape player controls
- [✓] Scroll bars

- [✓] Status bar

Toolbars
- [✓] Standard
- [✓] Edit
- [✓] Run Control
- [✓] Join/Split
- [✓] Body
- [✓] Points
- [✓] Joints
- [✓] Cnstr
- [ ] Simple

Close

Figure 2-4
Workspace dialog (Windows)

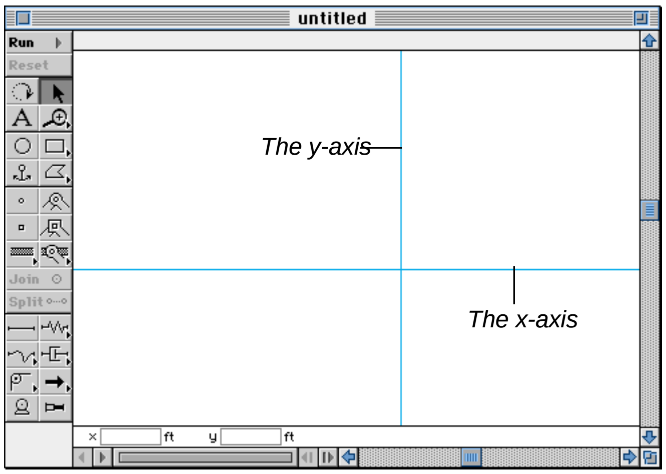


Figure 2-5
Empty workspace

2.3. Creating the Rod

This exercise requires a 4-foot long, 60-pound rod, which will be modeled as a thin rectangular body. It will be sized using the Geometry window, and its mass will be set using the Properties window. Working Model automatically calculates the moment of inertia of all objects as if they were two-dimensional plates of uniform density. The rectangle will be made thin so that its moment of inertia approximates that of a rod.

Drawing the Rod

To draw the rod:

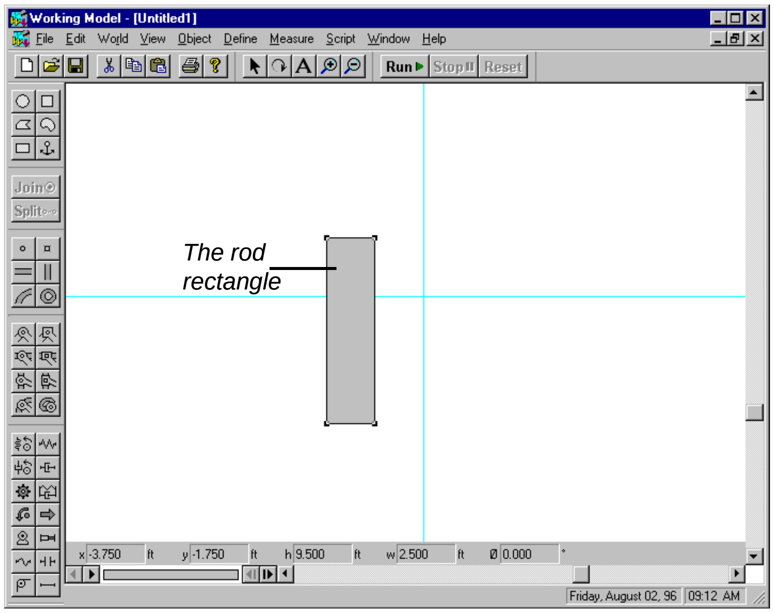
1. **Click the Rectangle tool in the Toolbar.**

   The Rectangle tool is selected.

2. **Position the pointer in the workspace and click to begin drawing. Move the mouse to size the rectangle. Click again to complete the rectangle.**

   A rectangle is created. The simulation window should look similar to Figure 2-6 .

Figure 2-6
Drawing the rod


Working Model - [Untitled1]
File Edit World View Object Define Measure Script Window Help
Run
Stop
Reset
Join
Split
The rod rectangle
x -3.750 ft y -1.750 ft h 9.500 ft w 2.500 ft ø 0.000 °
Friday, August 02, 96 09:12 AM

Sizing the Rod

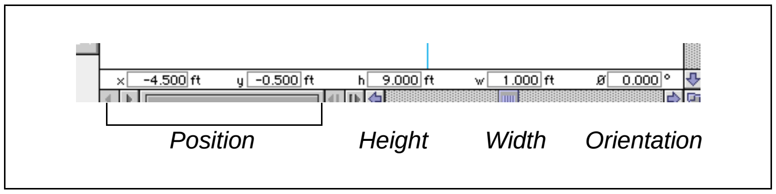
The rectangle must be 4 feet in length and must be thin to approximate the moment of inertia of a rod. Dimensions of 4 feet by 0.35 feet will be entered in the Coordinates bar. A mass of 60 pounds will be entered in the Properties window.

1. **Select the rod (if it is not already selected) by placing the pointer on the object and clicking.**

   Four square dots (called resize handles) appear at the corners of the rectangle to indicate that the object is selected.

Notice that the Coordinates bar shows the current position and dimensions of the rectangle. The position shown is that of the rectangle’s geometric center. The numbers are shown in the current unit system.

Figure 2-7
Coordinates bar for a rectangle



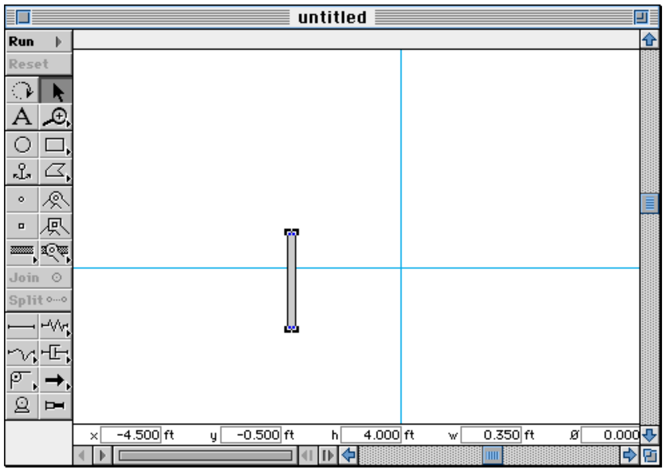
2. **Click the height field of the Coordinates bar (labeled h) and enter 4.0. Then press Tab.**

   The rod's height becomes 4 feet. Pressing tab selects the next field in the Coordinates bar (in this case, the width field).

3. **In the width field, enter the value 0.35 and press Return or Enter.**

   A value of 0.35 is used for the width so that the rectangle is thin enough to approximate the moment of inertia of a rod. The simulation window should resemble Figure 2-8.

Figure 2-8
The sized rod



Zooming In

The rectangle appears small after sizing. To make the rectangle (rod) appear larger:

1. **Click the Zoom In tool in the Toolbar.**

The Zoom In tool is selected, and the pointer changes to a magnifying glass marked with a plus sign.

2. **Place the pointer on or near the rectangle and click.**

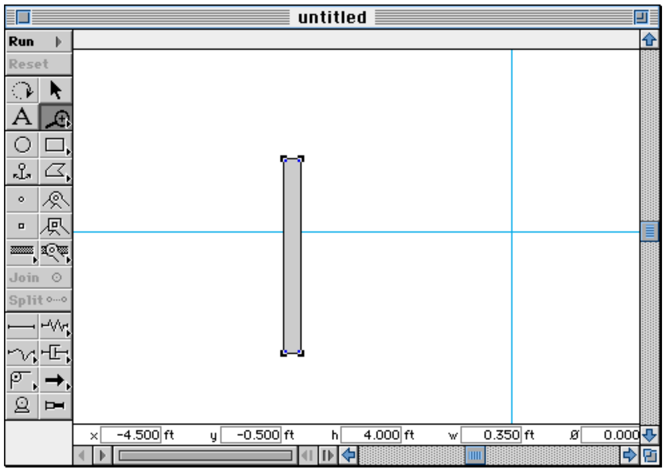
   The workspace is magnified by a factor of two with each mouse click.

   To zoom out while the Zoom In tool is selected, press the Shift key (a "-" will appear in the magnifying glass pointer) and click.

3. **Click the Arrow tool in the Toolbar or press the spacebar to deselect the Zoom In tool.**

   The pointer reverts to the standard arrow. The simulation window should resemble Figure 2-9.

Figure 2-9
The workspace after zooming in



x -4.500 ft y -0.500 ft h 4.000 ft w 0.350 ft ø 0.000

Setting the Weight of the Rod

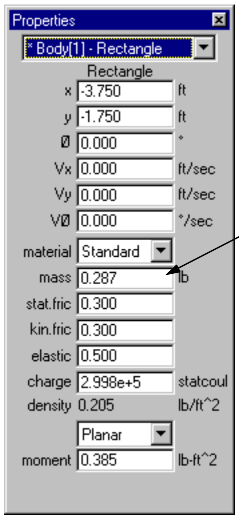
The rod in this exercise weighs 60 lbs. To set the rod's weight:

1. **Select the rod.**
2. **Choose Properties from the Window menu.**

The Properties window appears (Figure 2-10).The window can also be displayed by double-clicking on the object or by pressing Command+I (MacOS) or Control+I (Windows).

The Properties window shows various editable parameters of the selected object(s). Some of the parameters, such as position and orientation, are also shown in the Coordinates bar for quick access.

Figure 2-10
Properties window for a rectangle



x	-3.750	ft
y	-1.750	ft
Ø	0.000	°
Vx	0.000	ft/sec
Vy	0.000	ft/sec
VØ	0.000	°/sec
material	Standard	
mass	0.287	lb
stat.fric	0.300	
kin.fric	0.300	
elastic	0.500	
charge	2.998e+5	statcoul
density	0.205	lb/ft^2
	Planar	
moment	0.385	lb-ft^2

3. Enter the value 60 in the mass field.

2.4. Creating the Slot Joints

Joints in Working Model are created by “joining” elements with the Join command. In this exercise there are two slot joints. Slot joints are created by joining points to slots.

Finding Snap Points on the Rod

This exercise has two slot joints, one at each end of the rod. Each slot joint requires a point on the rod. To create the points:

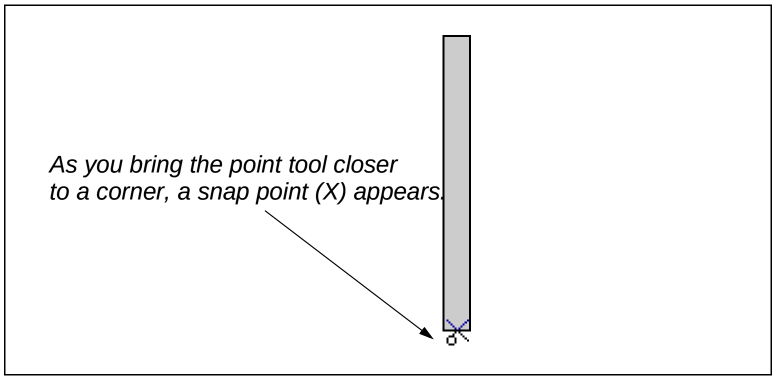
1. Double-click the Point tool in the Toolbar.

Double-clicking selects a tool for successive operations. On MacOS systems, the difference between single and double-clicking is indicated in the Toolbar by shading: a double-clicked item is dark grey, while a single-clicked item is light grey.

2. **Move the pointer over either end of the rectangle but do not click yet. Notice how a small X appears in some key locations.**

   The X symbol indicates the locations of Snap Points (Figure 2-11).

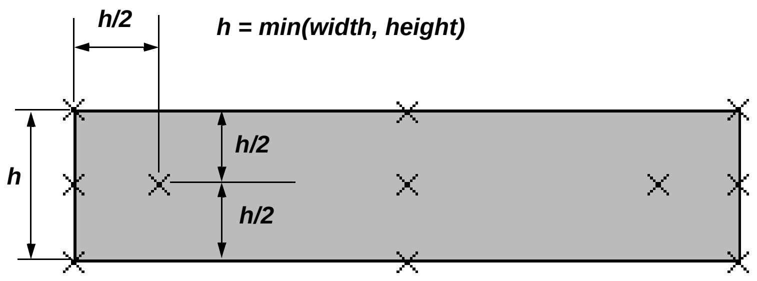
Figure 2-11
Finding snap points


As you bring the point tool closer to a corner, a snap point (X) appears.

Working Model allows you to attach points precisely at certain predefined positions on bodies, called *snap points.*

A rectangle in Working Model has 11 snap points (shown in Figure 2-12). Notice how snap points are arranged at midpoints and corners. Once an object is attached to a snap point, its position relative to the body is preserved even if the body is subsequently reshaped.

Figure 2-12
Snap Points for a rectangle



We now proceed to attach point elements to the rod.

Attaching Points to the Rod

We will attach a point element to each end of the rod.

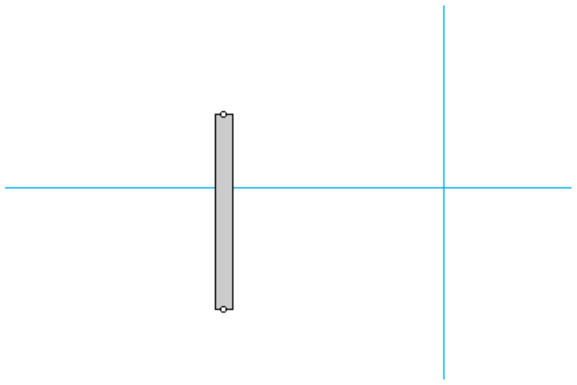
1. **Select the Point tool if you have not already done so.**

2. **Find the snap point at the center of the bottom end of the rod. When the snap point symbol appears, click to attach a point element.**

3. **Repeat the previous step for the top end of the rod.**

Your model should look like Figure 2-13.

Figure 2-13
Accurately positioned points at the ends of the rod



Naming Key Elements of the Model

Working Model automatically assigns a default name (such as “Rectangle” and “Point”) to each object you create. However, you will find it extremely helpful to assign more meaningful names to the key elements of your simulation. These names will come in very handy when you try to quickly locate a certain object using the Properties window, for example.

For now, we will assign names to the top and bottom point elements attached to the rod.

To assign a name to the point elements:

1. **Click the point element located at the top end of the rod.**

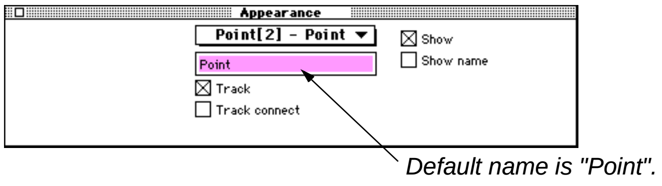
The point becomes highlighted.

2. **Choose Appearance from the Window menu.**

   The Appearance window appears (Figure 2-14).

The Appearance window provides control of how an object appears on the screen. The settings in this window pertain only to the appearance of the object; none of the settings in this window actually affect the results of a simulation.

Figure 2-14
Appearance window for a point



3. **Click the name field of the Appearance window (see Figure 2-14), and type Top Slot Pin.**

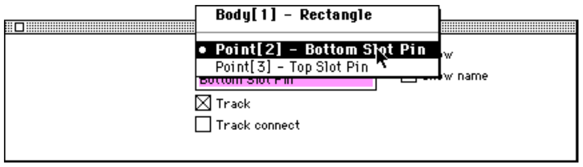
4. **Click the point element located at the bottom end of the rod.**

   Notice that the Appearance window automatically switches to display information for the bottom point element.

5. **Click the name field of the Appearance window, and type Bottom Slot Pin.**

We will later refer to these names. To see the list of all objects names, simply click the selection pop-up menu located in the Appearance window. This selection pop-up menu is also available in the Properties and Geometry windows.

Figure 2-15
Selection pop-up menu



Creating the Slots

Two slots are required for this exercise: one horizontal and one vertical. The slots will be created using the Slot tools.

To create the two slots:

1. **Click the Horizontal Slot tool in the Toolbar.**

2. **Bring the pointer near the origin and find its snap point (Figure 2-16). Click when the snap point is visible.**

   The horizontal slot is created, perfectly aligned with the x-axis.

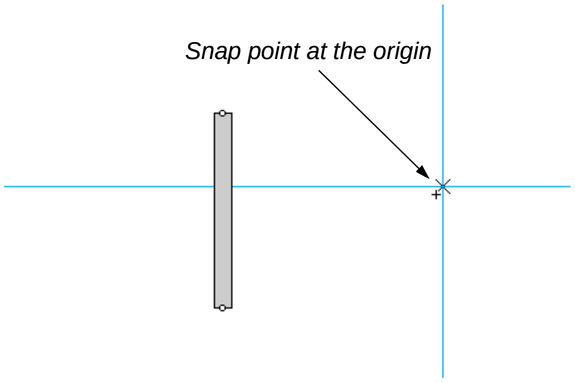


Figure 2-16
Snap point at the origin

3. **Click the Vertical Slot tool in the Toolbar.**

   On MacOS systems, the Vertical Slot tool is "hidden" in the Slot pop-up palette by default. Click and hold on the Horizontal Slot tool to bring the Slot pop-up palette in view (Figure 2-16).

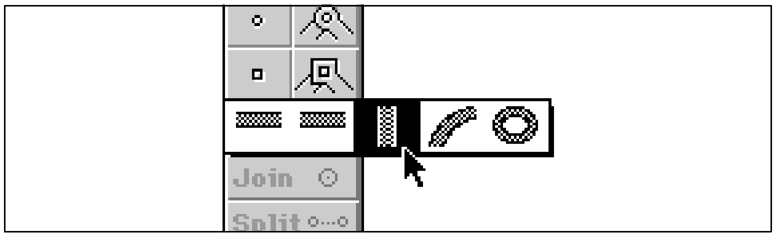


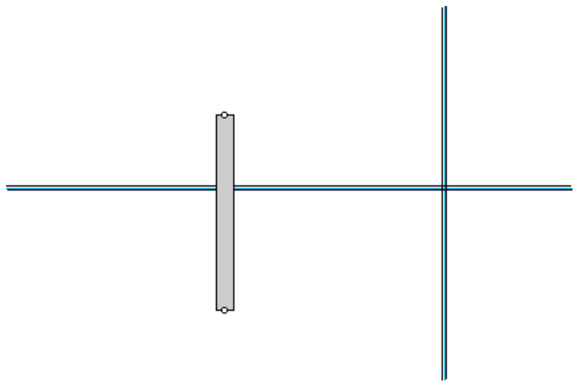
Figure 2-17
Slot pop-up palette (MacOS only)

4. **Find the Snap Point at the origin and click.**

The vertical slot is created, perfectly aligned with the y-axis.

The slots are now created on the background as shown in Figure 2-18 below.

Figure 2-18
Slots located on the x and y axes



Joining the Points to the Slots

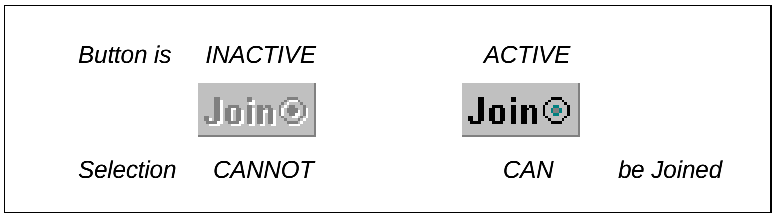
A joint is made by joining two primitive elements. When two elements are joined, objects move to satisfy the conditions of the joint. Joints never come apart, even when you drag bodies with the mouse. The Working Model Smart Editor™ allows you to drag bodies around while still satisfying all joints. In this exercise, a type of joint called a Slot Joint is created. A slot joint is made by joining a slot element and a point element. The rod requires two slot joints: one for the horizontal slot, and one for the vertical slot.

To join the points to the slots:

1. **Select the top point and, while holding the Shift key down, select the horizontal slot.**

   The word “Join” on the Join button now turns from gray to black to indicate that the two items can be joined (Figure 2-19).

Figure 2-19
Join button



2. **Click the Join button in the Toolbar.**

   The point is joined to the slot. The rod moves if necessary to satisfy the constraint. To separate the joint, you can select the slot or the point and click the Split button in the Toolbar.

   A portion of the rod may have moved out of view. If so, scroll the window so that the entire rod is visible.

3. **Select the bottom point and, while holding the Shift key down, select the vertical slot.**

4. **Click the Join button in the Toolbar.**

   Your model should resemble Figure 2-20.

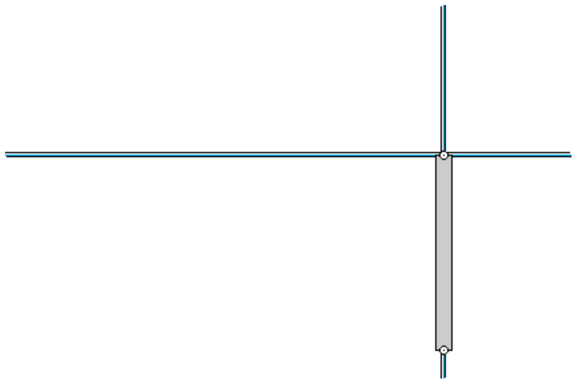


Figure 2-20
Rod with the points and slots joined

The Smart Editor will keep joints together during editing. To see this, de-select everything by clicking once on the background (or else points and slots may be dragged). Then try dragging the rod. The rod will stay constrained to the slots as it is dragged. When you are done dragging the rod, move it back to the approximate position shown in the problem description on the first page of this exercise.

2.5. Creating the Force

A 30 pound force is applied to the top of the rod. The force will be created with the Force tool. It will be sized and positioned using the Coordinates bar. The following steps demonstrate how to create the force and position it as stated in the problem.

To create the force:

1. **Click the Force tool in the Toolbar.**

   The Force tool is selected.

2. **Bring the pointer near the top of the rod and look for the Snap Point where the top slot pin is attached.**

   If necessary, see "Finding Snap Points on the Rod" on page 2–9 for review.

3. **When the snap point on the top slot pin is visible, click once and move the pointer horizontally to the left.**

   Observe that the force vector is sized according to the position of the mouse pointer.

4. **Click again to finish creating the force.**

   Your model should resemble Figure 2-21.

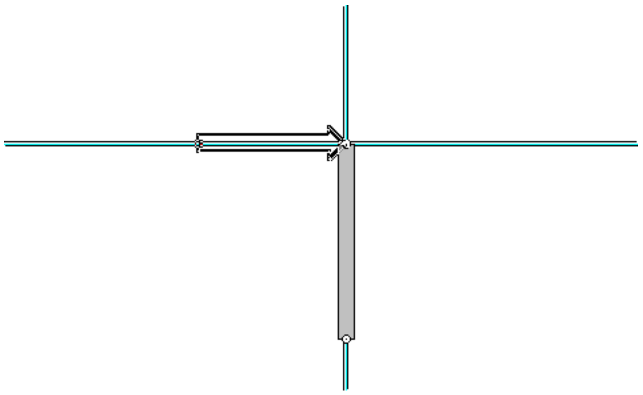
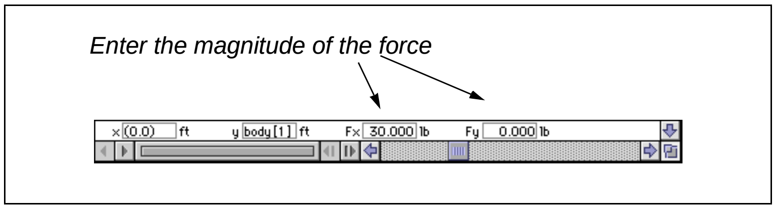


Figure 2-21
Force attached to the rod

5. **Click to select the force if it is not already selected.**

The force is highlighted to indicate that it is selected.

Figure 2-22
Coordinates bar for a force



6. **Click in the F_x field of the Coordinates bar and type 30.**

7. **Click in the F_y field of the Coordinates bar and enter the value 0.**

   The force now has the magnitude and direction stated in the problem description. The arrow representing the force now extends past the left edge of the simulation window to reflect the increased magnitude of the force. Do not attempt to scroll or zoom the simulation window to fit the force arrow; it will be resized later (see "Scaling the Vectors" on page 2–20).

2.6. Positioning the Rod

In this exercise, the initial rotation of the long axis of the rod is 60° counterclockwise from the positive x-direction. The rectangle must be rotated to match the exercise. The Rotate tool could be used to graphically rotate the rod; when precision is required, however, the value should be entered in the Properties window. To position the rod accurately:

1. **Click the rod to select it.**

2. **Click the Ø field (rotation) of the Coordinates bar and enter the value -30.**

   The rod will rotate 30° clockwise (Figure 2-23). In Working Model, positive rotation is measured counter-clockwise from the positive x-axis.

   Notice that the x and y positions of the rod change so that the two slot constraints remain satisfied.

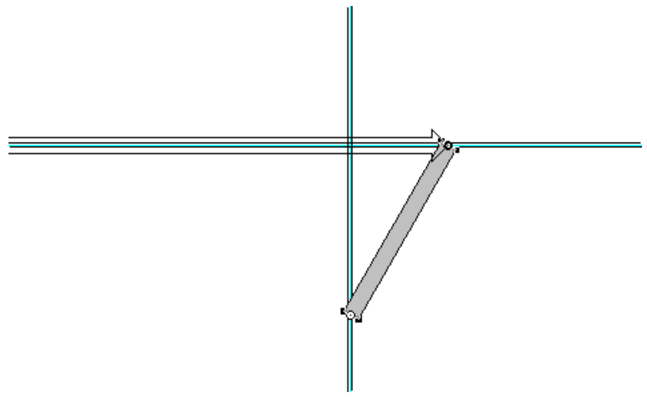


Figure 2-23
Rod after rotation

2.7. Running the Simulation

The simulation is now ready to run.

To run the simulation:

1. **Click the Run button in the Toolbar.**

   On MacOS systems, the Run button changes to a Stop button while the simulation is running.

   The rod oscillates up and down.

You must always reset the simulation before attempting editing. If you do not, the initial conditions of the simulation will be affected.

2. **Click the Reset button in the Toolbar.**

2.8. Measuring Properties from the Simulation

The initial forces on the joints and the initial angular acceleration of the rod must be measured. Working Model allows you to measure and represent physical properties such as force and acceleration using meters and vectors.

Displaying Vectors

The exercise asks for the initial force on both joints. You can display these forces as vectors for qualitative analysis. To display vectors:

1. **Choose Properties in the Window menu.**

   The Properties window appears.

2. **Click the selection pop-up menu and select "Top Slot Pin" (Figure 2-24).**

   The custom name you assigned in "Naming Key Elements of the Model" on page 2–11 comes in very handy in locating the point element. The pin becomes highlighted when selected.

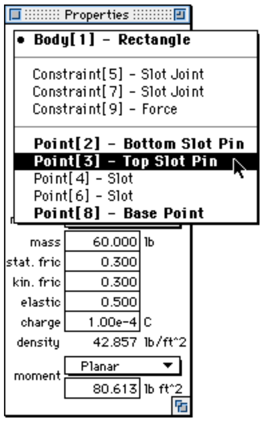


Figure 2-24
Finding theTop Slot Pin

3. **Choose Vectors from the Define menu and Total Force from the Vectors submenu (Figure 2-25).**

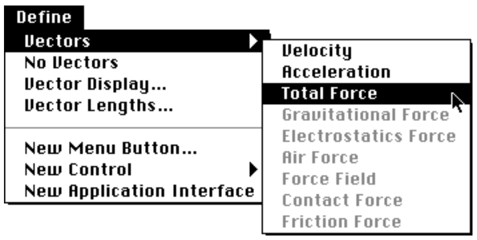


Figure 2-25
Choosing vectors

4. **Go back to the Properties window and select "Bottom Slot Pin".**

5. **Choose Vectors from the Define menu and Total Force from the Vectors submenu.**
6. **Run the simulation.**

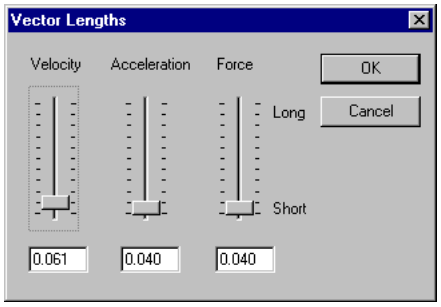
   Like the force object, the vectors do not fit on the screen.
7. **Click Reset.**

Scaling the Vectors

The vectors must be scaled to fit on the screen. To scale the vectors:

1. **Choose Vector Lengths… from the Define menu.**

   The Vector Length dialog (Figure 2-26) appears.



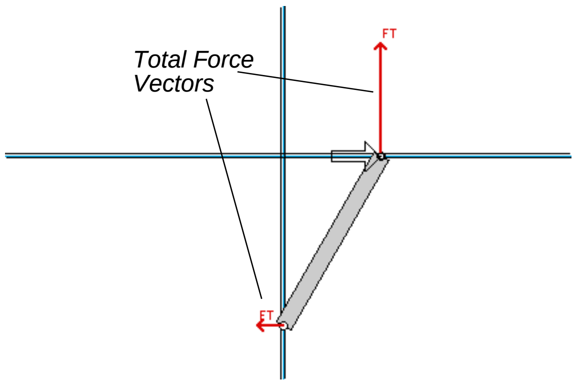
0.061
0.040
0.040

Figure 2-26
Scaling the vectors

2. **Click in the Force Vector field.**
3. **Enter a smaller value and press Tab to immediately see the change in the simulation window. Repeat until the vectors fit nicely into the window (try 0.0007). Click OK when done.**

Your model is now complete and should resemble Figure 2-27. Notice that changing the force vector scale affects the displayed length of the force object (attached to the top of the rod) as well.

Figure 2-27
Completed model



Displaying Digital Meters

Three digital meters are required for this exercise: two force meters for the slots and one angular acceleration meter for the rod. To create the slot force digital meters:

1. **Select the horizontal slot and choose Force from the Measure menu.**

   A force meter is created (Figure 2-28).

2. **Repeat for the vertical slot.**

   A meter can be moved to any location on the screen. Simply select the meter and drag it to a new location.

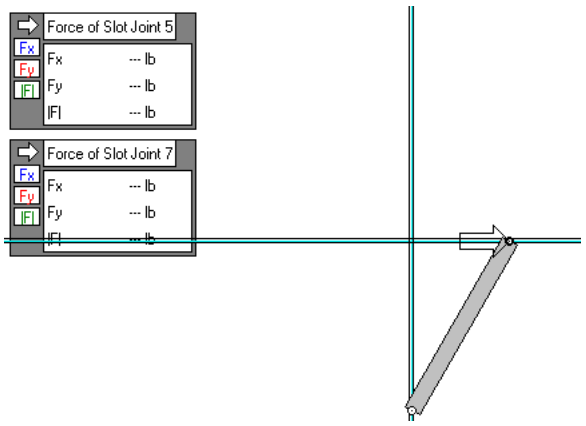


Figure 2-28
Force meters added to the simulation

To create the angular acceleration digital meter:

1. **Select the rod.**
2. **Choose Acceleration from the Measure menu, and Rotation Graph from the Acceleration submenu.**

   An x-y graph of the angular acceleration of the rod will appear (see Figure 2-29).

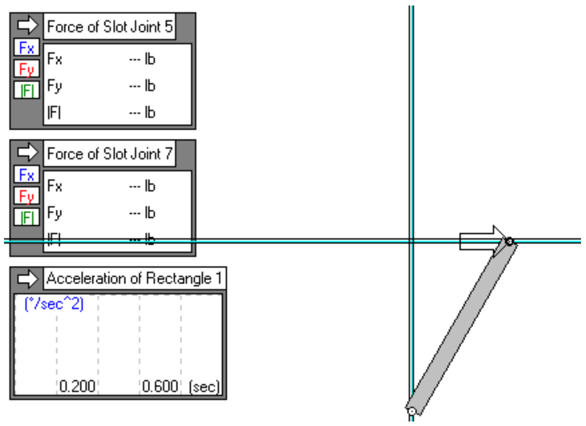


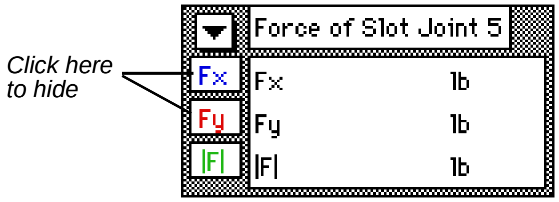
Figure 2-29
The x-y graph of the angular acceleration is displayed

Customizing the Meters

This exercise requires only the magnitudes of the forces on the slots. Thus, some of the properties displayed by the force meters should be hidden. For this example, the total force on the slots, |F|, is the only value of interest; F_x and F_y will be switched off. Also, if a numerical value for the angular acceleration is desired, rather than a graph, it too can be displayed. To modify meter displays:

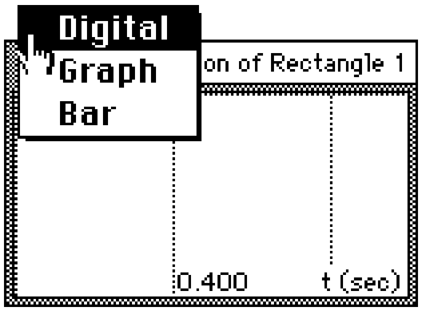
1. **Click the F_x and F_y buttons on the left side of the Force meters (see Figure 2-30).**

Figure 2-30
A digital force meter



2. **On MacOS systems, click and hold the arrow in the top left corner of the angular acceleration meter and choose the Digital option from the pop-up menu (see Figure 2-31). On Windows systems, click the arrow in top left corner of the meter. With each click, the meter type cycles from Digital, Graph, Bar, and Digital again.**

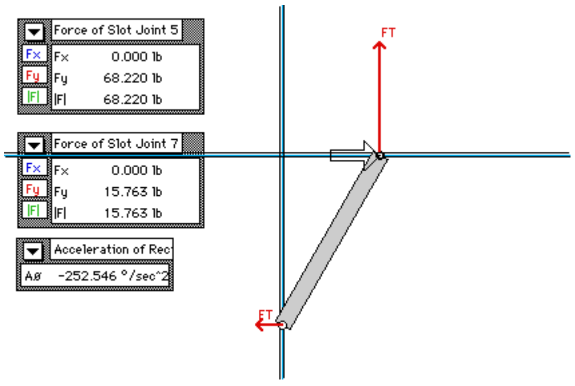
Figure 2-31
Changing the meter type



2.9. Checking the Answers

Congratulations! Exercise 2 is now completed. Run the simulation and reset to check the answers.

Your final screen should resemble Figure 2-32 below.



```
Force of Slot Joint 5
Fx     0.000 lb
Fy    68.220 lb
|F|   68.220 lb

Force of Slot Joint 7
Fx     0.000 lb
Fy    15.763 lb
|F|   15.763 lb

Acceleration of Rec
Aø   -252.546 °/sec^2
```

Figure 2-32
The final screen

Compare your answers with those below.

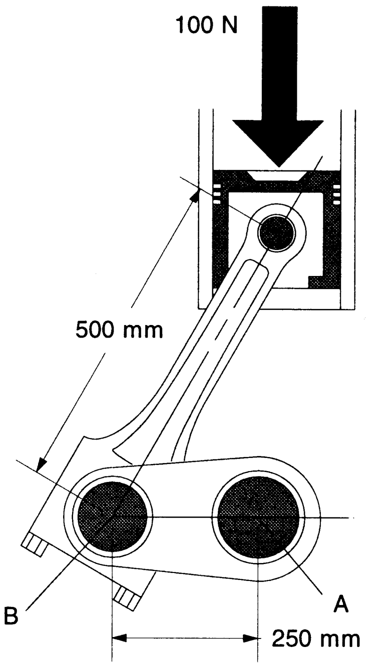
Force at point A................................≈ 16 lbs.

Force at point B................................≈ 68 lbs.

Angular Acceleration...................≈ -250 °/sec^2

E X E R C I S E   3

A Piston Engine



In the two cycle piston engine shown, explosive gases are ignited in the combustion chamber above the piston. The explosions apply a force of 100 N for the duration of every downward stroke. The engine is equipped with a speed limiting device (rev limiter) which prevents the rotational speed from exceeding a set value (red-line). The masses of the piston and connecting rod are 1 kg and 2 kg, respectively. The mass of the crankshaft-flywheel assembly is 35 kg. Given that the red-line of the engine is 35 rad/sec (340 rpm), determine the forces at the crankshaft bearing (point A) and connection rod (point B) bearing. Assume the crankshaft-flywheel assembly can be modeled as a circular disk.

Exercise 3 Concepts:

- Using equations and formulas to control a force
- Creating a keyed slot joint
- Displaying an x-y graph
- Increasing the accuracy of a simulation

3.1. Introduction

Engines that exceed the manufacturer's maximum speed (over-revving) may be subject to excessive wear and possible failure. To prevent over revving, internal combustion engines are often fitted with a device known as a rev-limiter. When an engine exceeds its red-line, rev-limiters interrupt the ignition system, slowing the engine down. Once the speed drops below the maximum, the ignition system is switched back on.

In this exercise you will model an internal combustion engine equipped with a rev-limiter. The engine has three bodies: the piston, the connecting rod and the crankshaft. The piston will be modeled by a square body. The connecting rod will be modeled by a rectangular body. The crankshaft will be modeled as a circular body. The bodies will be drawn, sized and joined to each other and the background. The piston's cylinder walls will be modeled with a keyed slot joint. The force of combustion will be modeled by a force attached to the top of the piston. The resulting forces at the bearings will be measured.

3.2. Setting Up the Workspace

For this exercise, three changes in the workspace will be made. First, for clarity, the x-y axes will be displayed. The unit of distance will also be changed from meters (default) to millimeters.

1. **If the x-y axes are not currently displayed, choose Workspace from the View menu and choose X,Y Axes from the Workspace submenu (MacOS) or the Workspace dialog (Windows).**

   The x-y axes provide a reference frame for building a simulation.

2. **Choose Numbers and Units… from the View menu.**

The Numbers and Units dialog appears.

3. **Click the More Choices button.**

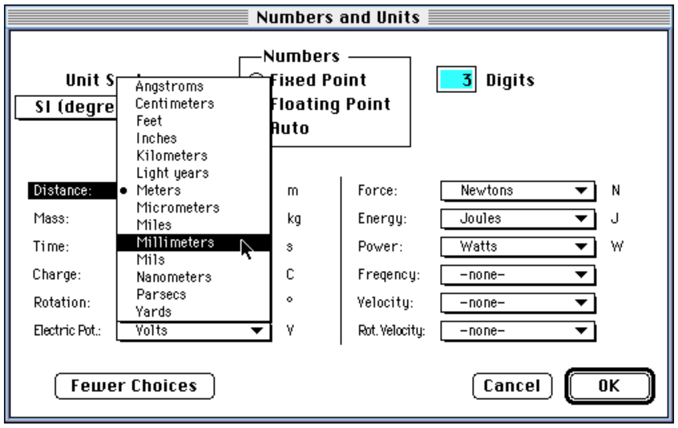
   The dialog box expands.

4. **Click and hold in the Distance field (Figure 3-1).**

   The pop-up menu appears.

5. **Choose millimeters from the Distance pop-up menu.**

Figure 3-1
Choosing millimeters as the unit of distance from the Numbers and Units dialog



3 Digits

Distance:	● Meters	m	Force:	Newtons	N
Mass:		kg	Energy:	Joules	J
Time:		s	Power:	Watts	W
Charge:		C	Freqency:	–none–	
Rotation:		°	Velocity:	–none–	
Electric Pot.:	Volts	V	Rot. Velocity:	–none–	

The unit of distance changes to millimeters.

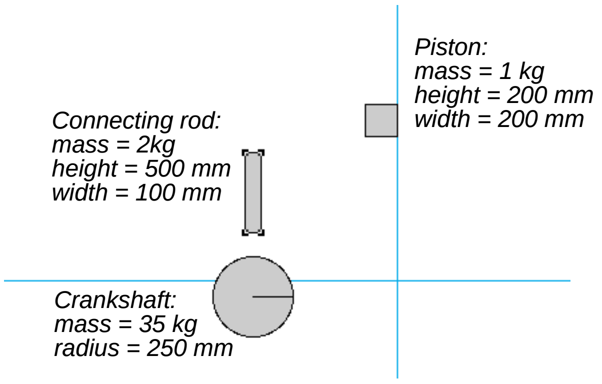
6. **Click in the Rotation field and choose Radians from the pop-up menu.**

7. **Click OK.**

3.3. Creating the Components

This exercise has three objects: a 2 kg connecting rod, a 1 kg piston, and a 35 kg crankshaft (see Figure 3-2 below). The connecting rod will be modeled by a thin rectangle 500 mm in length. Its width will be thin, 100 mm, so it closely resembles the actual connecting rod. The crankshaft, as stated in the exercise, is modeled as a circular disk with a 250 mm radius and a mass of 35 kg. A 200 mm square object will represent the piston. The objects will be created, sized and initialized in the following steps.

Figure 3-2
The bodies that will be created



Creating the Crankshaft

The crankshaft is represented by a circle with a radius of 250 mm and mass of 35 kg. These parameters will be set using the Geometry and Properties utility windows.

To draw the crankshaft:

1. **Click the Circle tool in the Toolbar.**
2. **Click once on the background. Move the mouse to expand the circle and click again to complete sketching.**

To set the mass of the crankshaft:

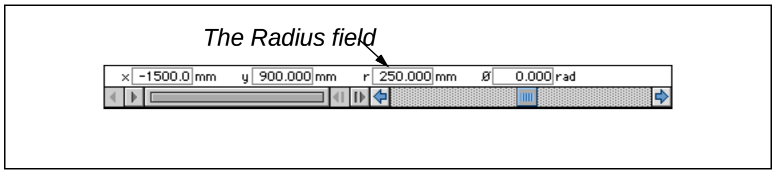
1. **Choose Properties from the Window menu.**
2. **Click the mass field and enter the value 35.**

To set the size of the crankshaft:

1. **Select the crankshaft if it not already selected.**
2. **Click the Radius field (labeled “r”) of the Coordinates bar and enter the value 250 (Figure 3-3).**

The Coordinates bar is located near the bottom of the window, just above the tape player controls.

Figure 3-3
Coordinates bar for a circle



x -1500.0 mm y 900.000 mm r 250.000 mm ø 0.000 rad

Zooming In

The circle will appear small after sizing. To make the rectangle easier to manipulate and appear larger, the workspace will be magnified using the Zoom tool. To zoom in:

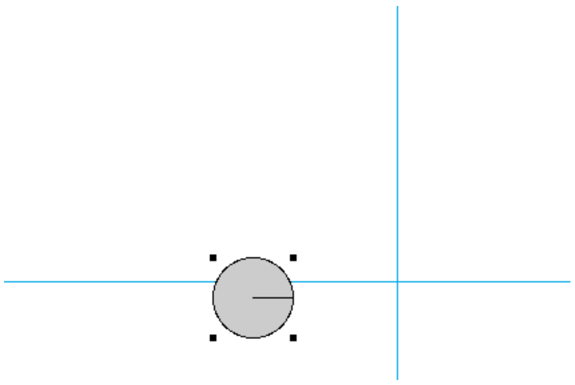
1. **Click the Zoom In tool in the Toolbar.**

2. **Click near the circle.**

   The objects in the window are magnified by a factor of two with each click of the mouse. To zoom out while the Zoom In tool is selected, hold down the Shift key (the magnifying glass pointer changes to "-") and click.

   Your screen should be similar to Figure 3-4 below.

Figure 3-4
The workspace after zooming in



3. **Click the Arrow tool in the Toolbar or press the spacebar to deselect the Zoom In tool.**

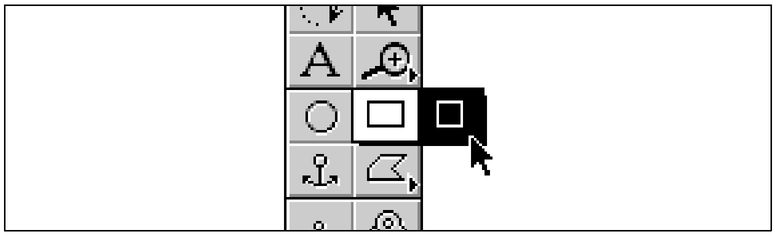
Creating the Piston

For this exercise, the piston will be modeled as an 200 mm square with the mass of 1 kg. The Square tool will be used to draw the piston, and the Properties window will be used to set its mass. To draw the piston:

1. **Choose the Square tool in the Toolbar.**

   On MacOS systems, the Square tool is "hidden" in the Rectangle/Square pop-up palette by default. Click and hold on the Rectangle tool to bering the pop-up palette in view (Figure 3-5 below).

Figure 3-5
Rectangle/Square pop-up palette (MacOS only)



2. **Click once on the background, drag to the right, and click again to complete sketching.**

   A square appears on the screen.

To set the size of the piston:

1. **Click in the Height or Width field of the Coordinates bar and enter 200.**

   Either field changes both the height and the width of the object so that it remains square.

To set the piston's mass:

1. **Select the piston if it not already selected.**
2. **Choose Properties from the Window menu.**
3. **Click in the mass field and enter the value 1.**

Creating the Connecting Rod

In this exercise, the connecting rod is represented by a rectangle drawn with the Rectangle tool and sized using the Geometry window. To approximate an actual connecting rod, the rectangle will be given a mass of 2 kg, a height of 500 mm, and a width of 100 mm.

To draw the connecting rod:

1. **Choose the Rectangle tool in the Toolbar.**

   On MacOS systems, the Square tool used above may have replaced the Rectangle tool in the Toolbar. If so, the Rectangle tool may be selected from the hidden pop-up palette by clicking and holding on the Square tool in the Toolbar.

2. **Click once on the background, drag to the right, and click again to complete sketching.**

   A rectangle appears on the screen.

To set the mass of the connecting rod:

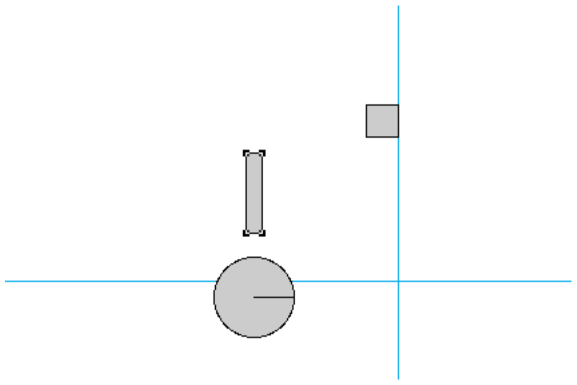
1. **Select the new rectangle if it not already selected.**
2. **Choose Properties from the Window menu.**
3. **Click in the Mass field and enter the value 2.**

To set the size of the rod:

1. **Click the Height field of the Coordinates bar and enter the value 500.**
2. **Click the Width field of the Coordinates bar and enter the value 100.**

   Your screen should resemble Figure 3-6.

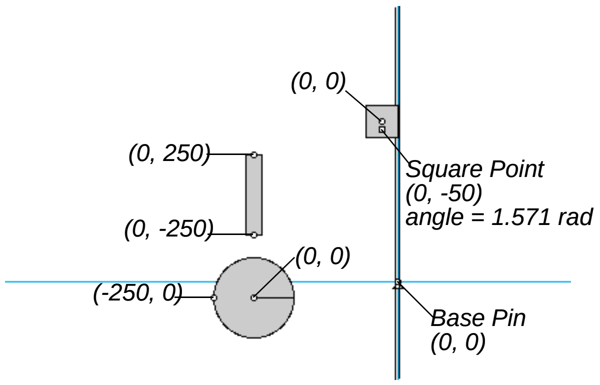
Figure 3-6
The piston, crankshaft and connecting rod



3.4. Creating the Points for Joining

The objects in this exercise are connected to each other and to the background (see Figure 3-7 below). The connections will be modeled by creating points and joining them. These points will be created with the Point tool and accurately positioned using the Properties window.

Figure 3-7
The points which will be created and their coordinates



Creating Points on the Connecting Rod

To create and position the points of the connecting rod:

1. Double-click the Point tool in the Toolbar.

Double-clicking selects a tool for successive operations. On MacOS systems, the difference between single and double-clicking is indicated in the Toolbar by shading: a double-clicked item is dark grey, while a single-clicked item is light grey.

2. **Place the mouse pointer over the connecting rod rectangle. Find the snap point at the top end of the connecting rod.**

   An X symbol appears at snap points. Find the snap point located at the midpoint of the top side of the connecting rod.

3. **To attach a point element, click when the snap point located at the top end is visible.**

   Observe that the point element is attached to the top end of the rod.

4. **In the same fashion, attach another point element to the bottom end of the connecting rod.**

Your screen should resemble Figure 3-8 below.

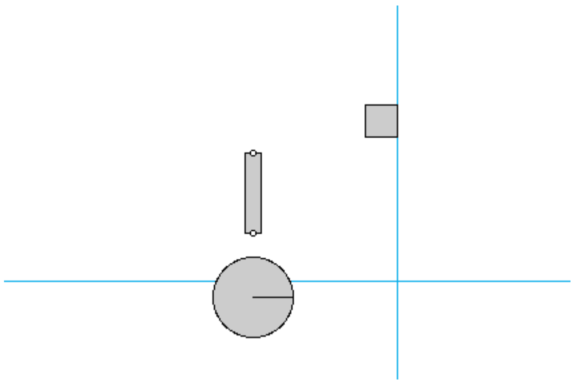


Figure 3-8
Connecting rod points in position

Attaching Points to the Crankshaft

The crankshaft needs two points: a center point representing the main bearing and an outer point representing the connecting rod bearing. These points will be accurately positioned at the center and at 250 mm from the center of the circle. To create and position the crankshaft points:

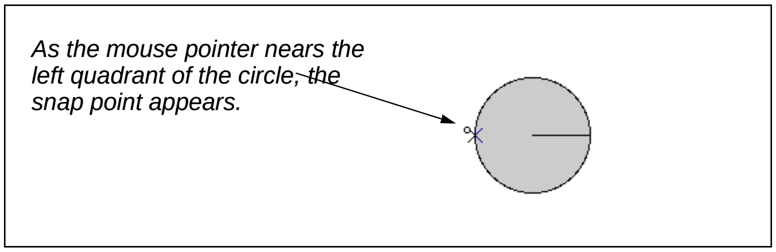
1. **Make sure the Point tool is still selected. Place the mouse pointer over the crankshaft. Find the snap point at the center of the circle.**

2. **Click when the snap point at the center is visible.**

   A point element is attached to the center of the circle.

3. **Place the mouse pointer near the left quadrant of the circle. Find the snap point at the left quadrant (Figure 3-9).**

Figure 3-9
Snap point at the left quadrant of the circle


As the mouse pointer nears the left quadrant of the circle, the snap point appears.

4. **Click when the snap point at the left quadrant is visible.**

   Another point element is attached to the left quadrant of the circle.

Attaching Points to the Piston

Two points need to be attached to the piston: a square point and a point. The square point will be joined to the slot on the background to create a keyed slot joint. The point will be joined to another point on the connecting rod to form a pin joint.

The square point will be positioned on the piston at the coordinates of (0, -50). The square point will be rotated 90°, so that when joined to the slot, the piston will not rotate. This is because vertical slots are defined as being rotated 90°; horizontal slots are defined as having 0° rotation. When a square point is joined to a slot, the rotations of the two objects are aligned.

The point will be positioned at (0, 0). To create the points:

1. **Click the Square Point tool.**

   The cursor turns into a square point.

2. **Click anywhere on the piston.**

   Since the square point will be precisely postion using the Coordinates bar below, it is not necessary to find a snap point.

3. **Click in the X field of the Coordinates bar and enter 0.**

   If you attached the square point on a snap point, the x-field of the Coordinates bar may contain a formula expression (such as `body[3].width` *or* `(0.0)`*). Simply overwrite the entire expression and enter 0.*

4. **Press the tab key to move to the Y field of the Coordinates bar, and enter -50.**

   Again, simply overwrite the y-field with the value -50.

5. **Choose the Properties in the Window menu.**

   The Properties window appears.

6. **In the angle field of the Properties window (marked by ø), enter the value 1.571.**

   The value of 1.571 radians is equivalent to 90°. The value will be displayed in a rounded form, but internally the correct value will be used.

   The point's y-position is set to -50 so it will not interfere with the connecting rod joint.

To create the point:

1. **Click the Point tool.**

2. **Place the pointer near the center of the square and find the snap point.**

3. **Click when the snap point at the center is visible.**

   The point is created at the center of the piston.

Attaching a Point to the Background

A point representing the main bearing of the crankshaft must be placed on the background. The point can be placed anywhere on the background; for clarity, it will be placed at the origin (0, 0). To create and position the point:

1. **Click the Point tool.**

2. **Place the mouse pointer near the origin. Find the snap point that appears at the origin.**

   Make sure that no body is obstructing the origin. The snap point does not appear if the origin is covered by a body.

3. **Click when the snap point at the origin is visible.**

   Verify the position of the point by noting the (x, y) position in the Coordinates bar; it should be (0, 0).

We now name this point element so that it becomes easier to select later.

4. **Choose Appearance from the Window menu.**

   The Appearance window appears.

5. **In the name field (where it says Point), type Base Pin.**

   Since we will not be needing the Appearance window for the rest of the exercise, close the window by: (Windows) clicking the box marked "X" at the top-right corner of the window, or (MacOS) clicking the box at the top-left corner of the window.

Attaching a Slot to the Background

The cylinder walls of the engine will be modeled with a keyed slot joint. The joint will be made by joining the square point on the piston to a slot attached to the background. The slot can be located anywhere on the background as long as it is located on the crankshaft's centerline (the y-axis for this model).

To create and position the slot on the background:

1. **Choose the Vertical Slot tool in the Toolbar.**

   On MacOS systems, the Vertical Slot tool may be "hidden" in the Slot pop-up palette. Click and hold on the Slot tool in the Toolbar to bring the pop-up palette in view.

2. **Place the pointer near the point element attached to the origin. Find the snap point.**

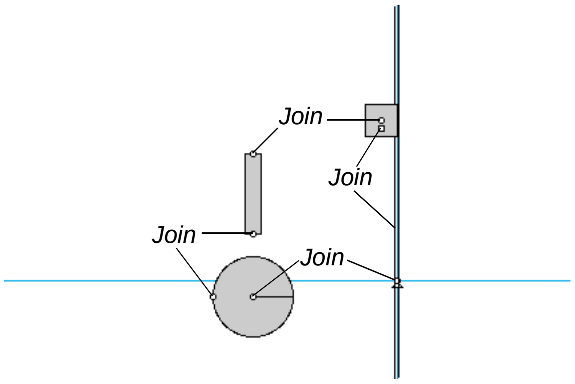
3. **Click when the snap point is visible.**

   A vertical slot appears.

3.5. Creating Joints

In this exercise there are four joints (see Figure 3-10). The joints will connect: the piston to the slot, the piston to the connecting rod, the connecting rod to the crankshaft, and the crankshaft to the main bearing. The following steps will demonstrate how to make these joints.

Figure 3-10
The points which will be joined



Joining the Piston to the Slot

In this example, the piston slides in a cylinder. The cylinder will be modeled as a keyed slot joint. The joint is created by joining the square point on the piston to the slot. To create the keyed slot joint:

1. **Select the square point on the piston, and while holding down the Shift key, select the slot.**

   The word "Join" of the Join button in the Toolbar changes from gray to black indicating the two points can be joined.

2. **Click the Join button in the Toolbar.**

   The piston moves to the slot (see Figure 3-11).

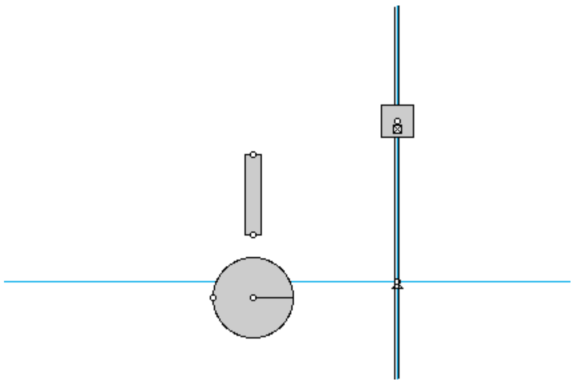


Figure 3-11
Piston joined to the slot

Joining the Crankshaft to the Point on the Background

The crankshaft main bearing is modeled as a pin joint. The pin joint will be created by joining the point at the center of the circle to the point at the origin.

To join the crankshaft to the origin:

1. **Choose Properties in the Window menu.**

2. **From the selection pop-up menu located at the top of the Properties window, choose Base Pin (Figure 3-12).**

   The Base Pin is now selected.

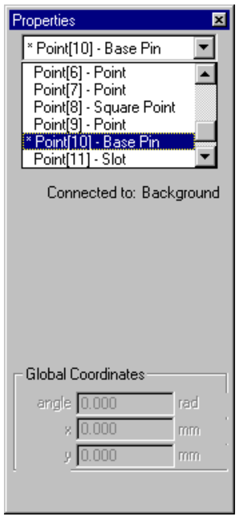


Figure 3-12
Selecting an object in the Properties window

3. **Hold the Shift key down and select the center point on the crankshaft circle.**

4. **Click the Join button.**

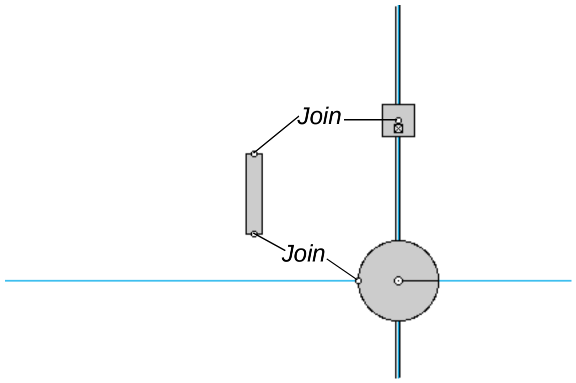
   The crankshaft circle moves to the origin.

To see how the circle representing the crankshaft is constrained, bring the mouse pointer over the circle, hold down the mouse button, and drag the circle. The circle can rotate but is fixed to the Base Pin.

Joining the Components

The piston, connecting rod and crankshaft will now be joined together (see Figure 3-13). The following steps describe how to join them.

Figure 3-13
Joining the components



To join the piston to the connecting rod:

1. **Select the (round) point on the piston and, while holding the Shift key down, select the top point on the connecting rod.**
2. **Click the Join button.**

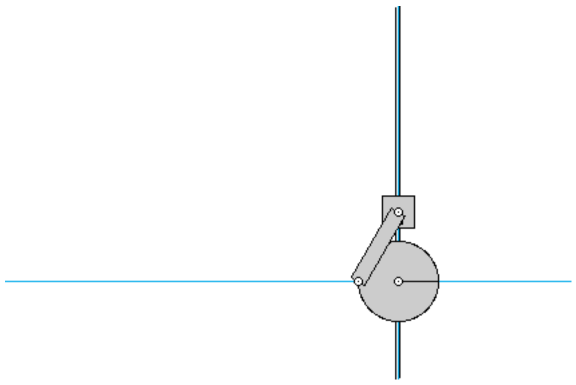
   The two objects will come together.

To join the connecting rod to the crankshaft:

1. **Select the bottom point on the connecting rod and, while holding the Shift key down, select the remaining (left) point on the crankshaft.**
2. **Click the Join button.**

   Your screen should resemble Figure 3-14 below.

Figure 3-14
The completed engine



The Smart Editor keeps joints together during editing. To see this, try dragging the connecting rod. The connecting rod will stay joined while it is dragged and the crankshaft and piston will move accordingly. When you are done dragging the rod, move it back to the approximate position shown in the problem description on the first page of this exercise.

Preventing a Collision

In Working Model, objects which are directly connected to each other never collide. Notice that the piston and crankshaft may overlap when they are dragged during editing. Since the two objects are not directly connected, they will collide when you run the simulation. To prevent the collision, the **Do Not Collide** command will be used.

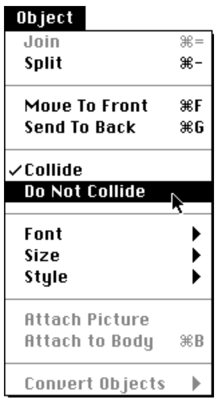
To prevent a collision:

1. **Select the piston and, while holding the Shift key down, select the crankshaft.**

2. **Click and hold on the Object menu title in the menu bar (Figure 3-15).**

   Notice that there is a checkmark beside "Collide", indicating that the pair of selected objects will collide when you run the simulation.

Figure 3-15
Editing the collision property of a pair of bodies



3. **Choose Do Not Collide in the Object menu.**

3.6. Creating the Force

A 100 N force is applied at the top of the piston in the negative y-direction (down). The force is only active while the piston is traveling with a negative y velocity. The force will be drawn using the Force tool and its magnitude and location will be set using the Properties window.

Drawing the Force

The force will be drawn in the downward direction using the Force tool. To draw the force:

1. **Click the Force tool in the Toolbar.**
2. **Place the pointer near the midpoint of the top end of the piston. Find the snap point.**
3. **Click when the snap point is visible, drag the mouse upward, and click again to create the force.**

   A force attached to the piston appears (Figure 3-16). Do not be concerned with the exact direction or magnitude of the force for now.

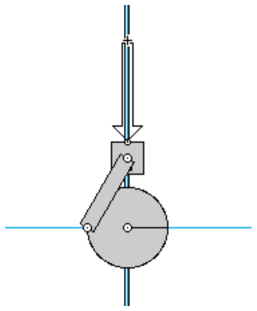


Figure 3-16
Creating a force

Sizing the Force

The force's magnitude and direction will be set using the Coordinates bar. To size the force:

1. **Click the force vector to select the force.**

2. **Click the F_y field of the Coordinates bar and type -100.**

3. **Click the F_x field of the Coordinates bar and type 0.**

   The force vector changes in length to reflect its new magnitude.

Timing the Force

The ignition timing of the engine must be simulated. The problem states that the force is active only on the downward stroke (*i.e.*, when the piston has a negative velocity). The rev-limiter cuts off combustion when the rotational velocity is greater than 35 radians/second. To simulate these conditions, the “and (a,b)” function will be incorporated with the following formulas.

To determine when the piston has a negative velocity use the following formula:

```
body[c].v.y < 0
```

where ***c*** is the piston's object id number. To determine the piston’s id number, place the cursor over the piston and read the Status bar. On MacOS systems, the Status bar is located at the top of the simulation window; on Windows systems, the Status bar is located at the bottom of the window.

To determine when the engine’s speed is greater than 35 rad/s, use the following formula:

```
body[d].v.r < 35
```

where ***d*** is the crankshaft's object id number. Use the Status bar as above to determine the crankshaft’s id number.

The complete equation is:

```
and(body[c].v.y < 0,body[d].v.r < 35)
```

The complete equation will be placed in the Active When field in the force’s Properties window. Gravity will provide the initial downward velocity to start the engine.

To set the timing of the force:

1. **Select the force.**
2. **Choose Properties from the Window menu.**
3. **Click in the Active When field at the bottom of the Properties window. On Windows systems, you must first un-check the Always button before clicking in the Active When field.**
4. **Type the following formula (see Figure 3-17):**

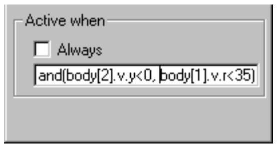
   and(body [***c***].v.y <0,body[***d***].v.r< 35)

   *where **c** and **d** are the object id numbers of the piston and crankshaft respectively.*

   The force's rev-limiter equation is designed for counter-clockwise (positive) engine rotation. It is important that the engine be "started" in a position similar to the figure on the first page of this section. If you want a rev-limiter which works in both directions try incorporating the abs() function into the formula (see the Working Model User's Manual *for more information on functions).*

   You can increase the width of the "Active when" field to view the entire equation by resizing the Properties window.

Figure 3-17
Active When field in the Properties window for a force



and(body[2].v.y<0, body[1].v.r<35)

The simulation is ready to run.

5. **Click the Run button in the Toolbar.**

   The engine runs. A warning may appear if you run the engine long enough; you can ignore the warning for now.

6. **When you are ready to continue, click the Reset button in the Toolbar.**

3.7. Measuring Properties from the Simulation

A graph in conjunction with a digital meter is a nice way to illustrate the rotational speed of the engine's crankshaft. Two digital meters will be used to display the forces on the bearings. Follow the steps below to create these output devices.

Displaying a Graph

The angular velocity of the crankshaft will be displayed by a graph. To create the graph:

1. **Select the crankshaft circle.**

   Four square "handles" will appear around the circle indicating that the circle is selected.

2. **Choose Velocity from the Measure menu and Rotation Graph from the Velocity submenu.**

   A meter resembling Figure 3-18 appears.

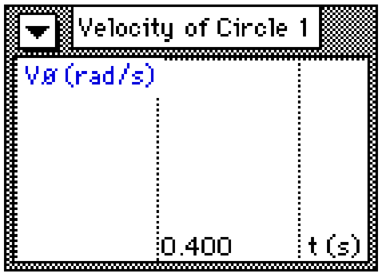


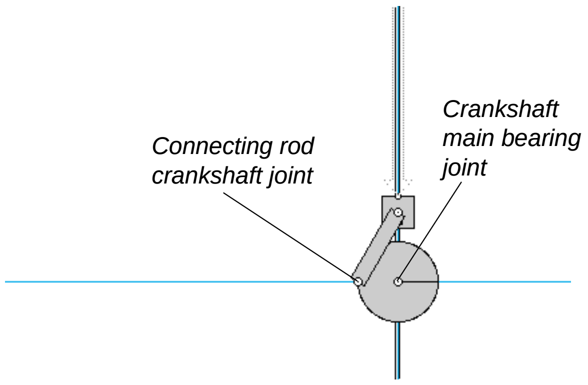
Figure 3-18
A velocity graph

Displaying Digital Force Meters

To display the digital force meters:

1. **Select the crankshaft-main bearing joint (see Figure 3-19).**

Figure 3-19
The forces to be measured


Connecting rod crankshaft joint
Crankshaft main bearing joint

2. **Choose Force from the Measure menu.**

   A digital meter appears.

3. **Repeat for the connecting rod-crankshaft joint (see Figure 3-19).**

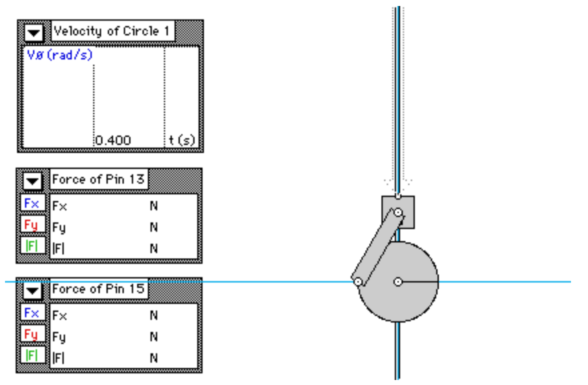
   A second digital meter appears. You may need to move the meters.

To move the meters:

1. **Select a meter.**
2. **Drag it to any position you wish.**

   Your window should resemble Figure 3-20.

Figure 3-20
The completed workspace



3.8. Running the Simulation

Run the simulation. Notice that the angular velocity increases until it reaches 35 rad/s. This is so because we have turned off the force above 35 rad/s, just as a rev limiter turns off the ignition of an engine above a certain speed.

Modifying the Graph Display

You may notice that the axis labels occlude the numeric labels along the tickmarks of the plot. You can modify the display options using the Appearance window.

For example, to turn off axis labels so that the numeric labels clearly appear:

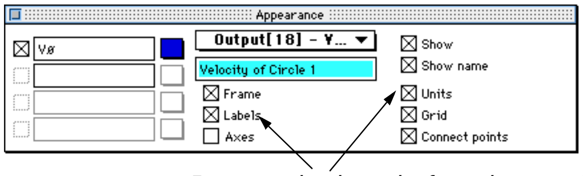
1. **Select the velocity meter on the screen.**
2. **Choose Appearance in the Window menu.**

   Alternatively, you could press Control+J (on Windows systems) or Command+J (on MacOS systems) to open the window.

3. **Turn off the options titled “Labels” and “Units” (Figure 3-21)**

   You may wish to try modifying other options and observe the effects on the graph meter.

Figure 3-21
Appearance window for a meter


Appearance

☒ Vø

Output[18] – Y...

Velocity of Circle 1

☒ Frame
☒ Labels
☐ Axes

☒ Show
☒ Show name
☒ Units
☒ Grid
☒ Connect points

Remove check marks from these options.

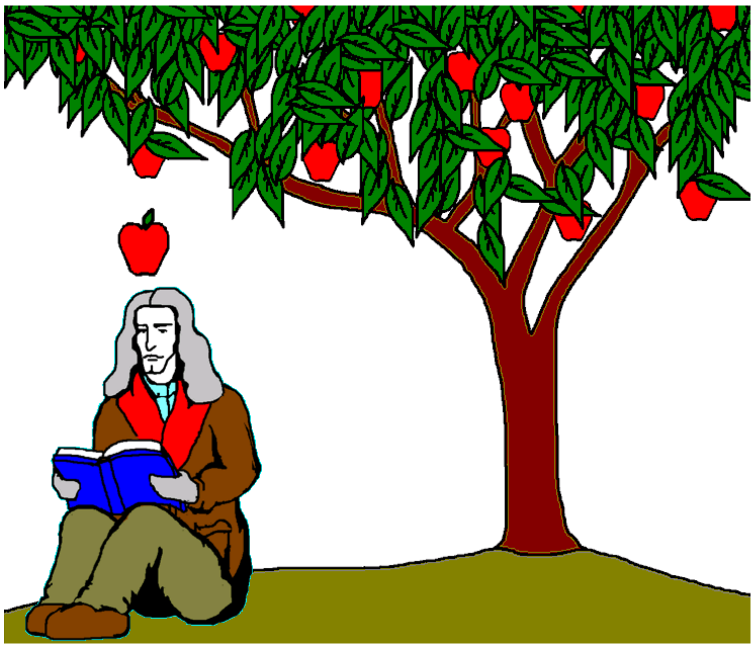
If you wish to show the meter coordinate axes, click the check box labeled Axes.

Modifying the Simulation

Try modifying the masses of the crankshaft and connecting rod and notice how quickly red-line is reached. Also try changing the length of the connecting rod by repositioning one of the joints.

EXERCISE 4

Making Visually Appealing Models



Concepts for Exercise 4:

- Changing the appearance of the workspace
- Adding pictures to the background
- Attaching pictures to a body
- Adding text to a simulation
- Other visual tips

4.1. Introduction

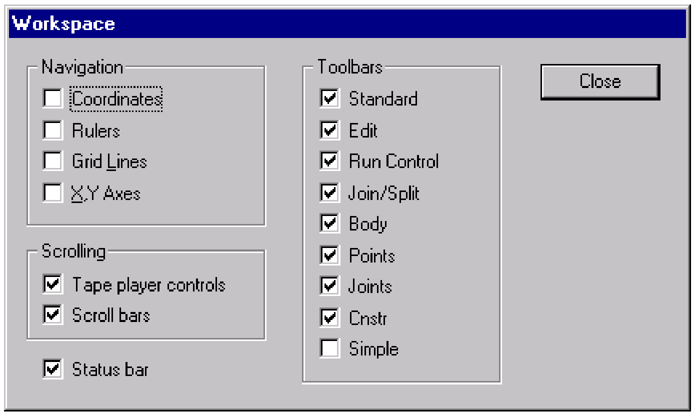
In this exercise, you will re-create the scene where Sir Isaac Newton discovered the fundamental law of motion **F**=m**a**. The simulation consists of an apple falling from an apple tree and hitting Newton on the head. After a few seconds, the text **F**=m**a** appears in large bold blinking letters.

4.2. Setting Up the Workspace

1. **From the View menu, select Workspace...**

   The Workspace dialog appears (Figure 4-1). From this dialog box, you can control the look-and-feel of the workspace.

Figure 4-1
Workspace dialog



Workspace

Navigation
Coordinates
Rulers
Grid Lines
X,Y Axes

Scrolling
[✓ Tape player controls]
[✓ Scroll bars]

[✓ Status bar]

Toolbars
[✓ Standard]
[✓ Edit]
[✓ Run Control]
[✓ Join/Split]
[✓ Body]
[✓ Points]
[✓ Joints]
[✓ Cnstr]
Simple

Close

2. **Under the Navigation frame, uncheck the boxes labeled Coordinates, Rulers, Grid Lines, and X,Y Axes.**

   Notice that the coordinates, rulers, grid lines, and axes disappear from the workspace as soon as you uncheck the boxes.

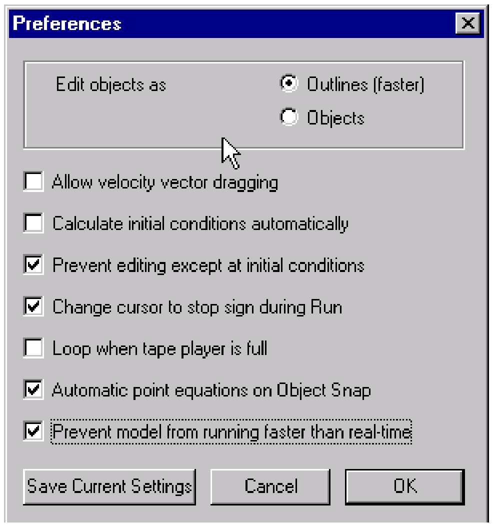
   Try checking and unchecking the other boxes and note the changes in the workspace, scroll bars, status bar, and toolbars.

3. **Click [Close].**

4. **From the World menu, select Preferences.**

   The Preferences dialog appears (Figure 4-2).

Figure 4-2
Preferences dialog


Preferences

Edit objects as [✓ Outlines (faster)] Objects

☐ Allow velocity vector dragging

☐ Calculate initial conditions automatically

☑ Prevent editing except at initial conditions

☑ Change cursor to stop sign during Run

☐ Loop when tape player is full

☑ Automatic point equations on Object Snap

☑ Prevent model from running faster than real-time

5. **Check the box labeled “Prevent model from running faster than real-time” and click [OK].**

4.3. Adding a Picture of the Apple Tree

1. **Using the Windows Explorer, go to the directory where Working Model is installed, for example, D:\Program Files\Working Model. Navigate to the sub-directory called Picture Library and then its sub-directory NewtonAndApple.**

2. **Double-click on the file "Tree.bmp."**

   This opens a bitmap picture of the tree in a program such as Microsoft Paint.

3. **Copy the picture to the clipboard. In Microsoft Paint, this is accomplished by clicking on the Edit menu and clicking on Select All, then clicking on the Edit menu and clicking on Copy.**

4. **To paste the picture of the tree into a Working Model document, select Paste from Working Model’s Edit menu.**

5. **Click and drag the Tree to the middle of the workspace. It should resemble Figure 4-3.**

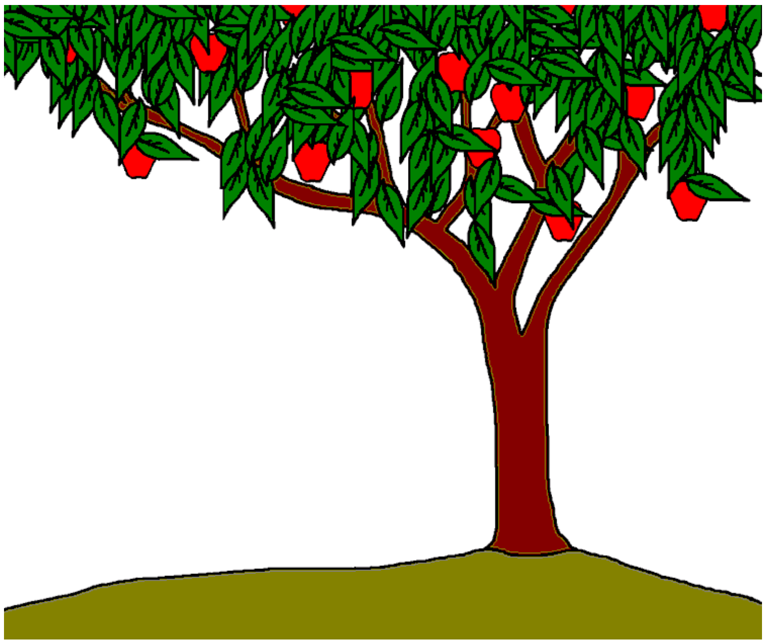


Figure 4-3
Tree as background

4.4. Adding a Picture of an Apple

1. **Double-click on the file Apple.bmp. It is located in the directory NewtonAndApple, the same directory where Tree.bmp is located.**

   This opens a bitmap picture of the apple in a program such as Microsoft Paint.

2. **Select and copy the picture of the apple.**
3. **To paste the picture of the apple into the Working Model document, select Paste from Working Model's Edit menu.**

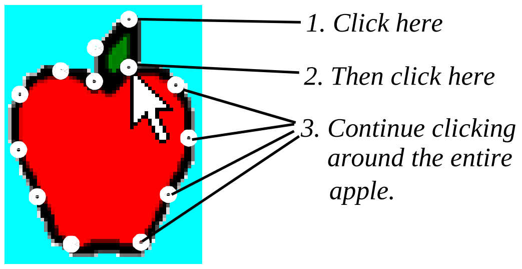
4.5. Creating a Body that Looks Like an Apple

1. **Select the Curved Polygon tool.**

2. **Click on any point on the boundary of the picture of the apple. Move your mouse clockwise along the boundary and single-click again. Keep single-clicking on points along the picture until you have traced the entire apple. (See Figure 4-4.)**

   Notice that with every click, you create a segment of the curved polygon.

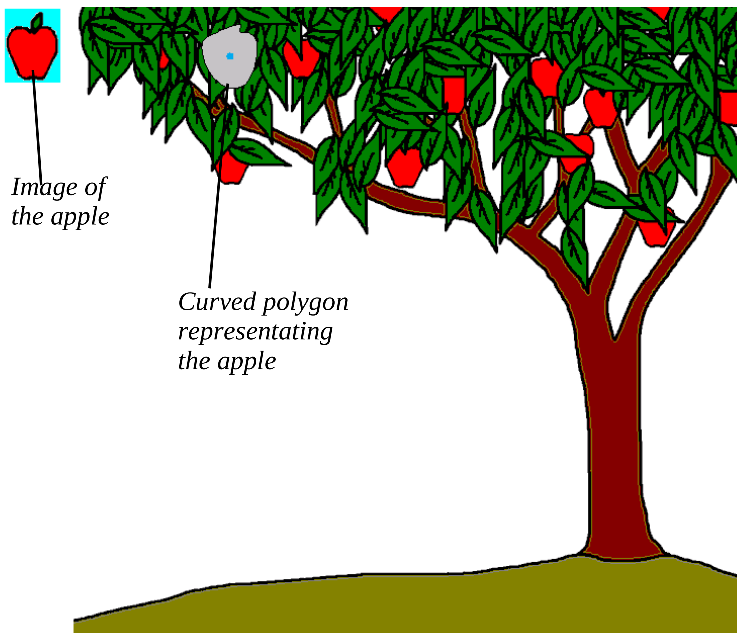
Figure 4-4
Tracing the apple



3. **Double-click when you return to the starting point.**

4. **Click and drag the curved polygon to the upper left-hand side of the tree. The workspace should resemble Figure 4-5.**

Figure 4-5
Putting the apple in the tree



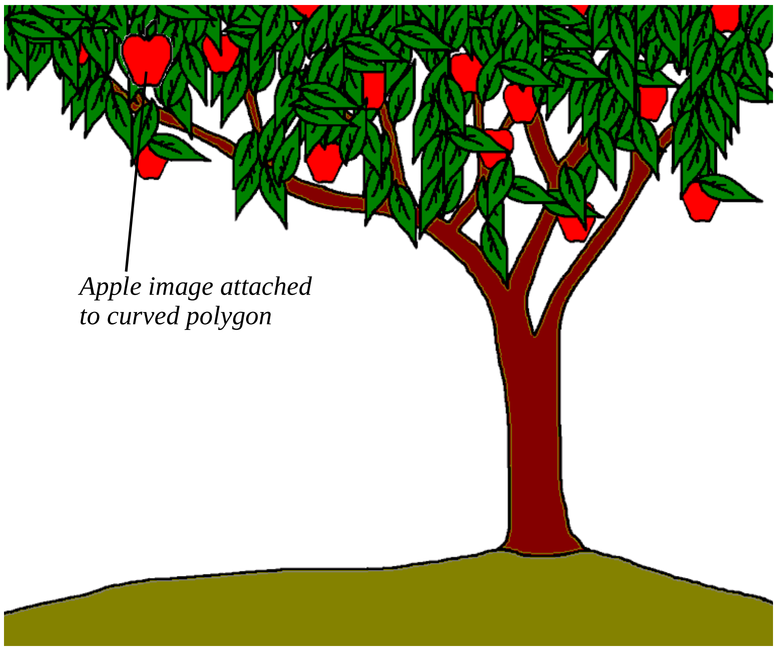
To attach the picture of the apple to the curved polygon:

1. **Click and select the picture of the apple, then hold down the [Shift] key while you click on the curved polygon that represents the apple.**

2. **Select Attach Picture from the Object menu.**

   Notice that the picture of the apple is attached to the curved polygon. The workspace should resemble Figure 4-6.

Figure 4-6
Tree and apple



4.6. Adding a Velocity Vector to the Apple

1. **Click on the Apple.**

2. **Select Vectors from the Define menu. Select Velocity from the Vectors submenu.**

3. **Run the simulation.**

Notice that the apple falls from the tree and the velocity vector elongates as it falls faster.

4. **Click Stop and Reset.**

4.7. Adding a Picture of Sir Isaac Newton

1. **Double-click on the file Newton.bmp. It is located in the directory NewtonAndApple, the same directory where Tree .bmp and Apple.bmp are located.**

   This opens the bitmap picture of Newton in a program such as Microsoft Paint.

2. **Select and copy the picture of Newton.**
3. **To paste the picture into Working Model document, select Paste from Working Model's Edit menu.**

4.8. Creating a Body that Resembles Sir Isaac Newton

1. **Select the Curved Polygon tool.**
2. **Click on any point on the boundary of the picture of Newton. Move your mouse clockwise along the boundary and single-click again. Keep single-clicking on points along the picture until you have traced the entire picture. (See Figure 4-7.)**

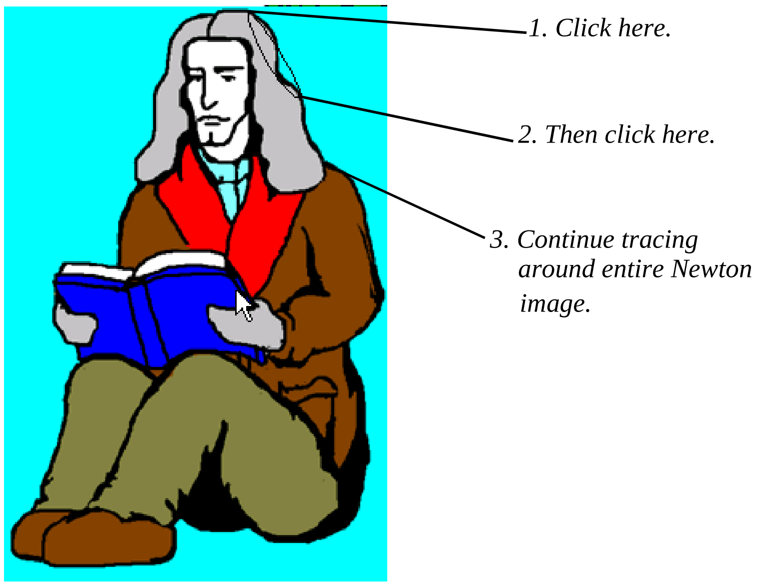


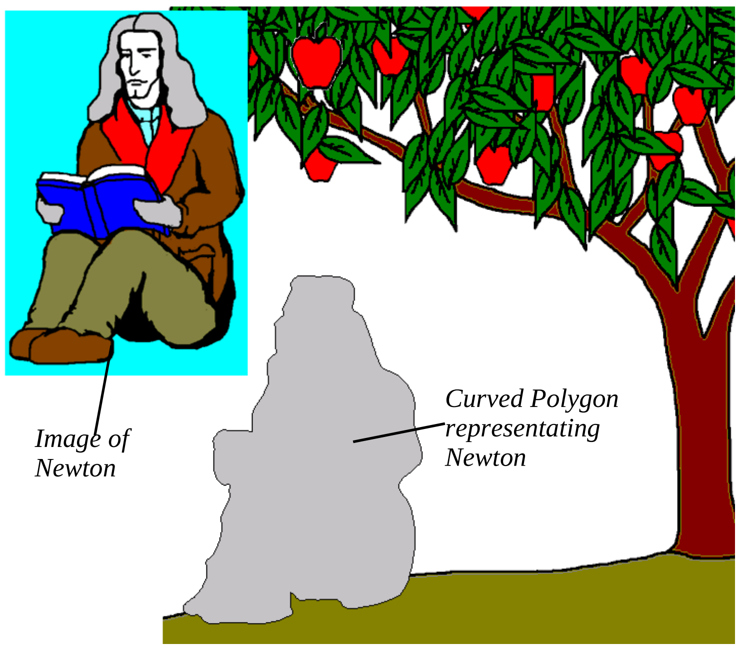
Figure 4-7
Tracing Newton

3. **Double-click when you return to the starting point.**
4. **Click and drag the curved polygon to the lower left-hand side of the tree.**

   The workspace should resemble Figure 4-8.

Figure 4-8

Putting Newton under the apple



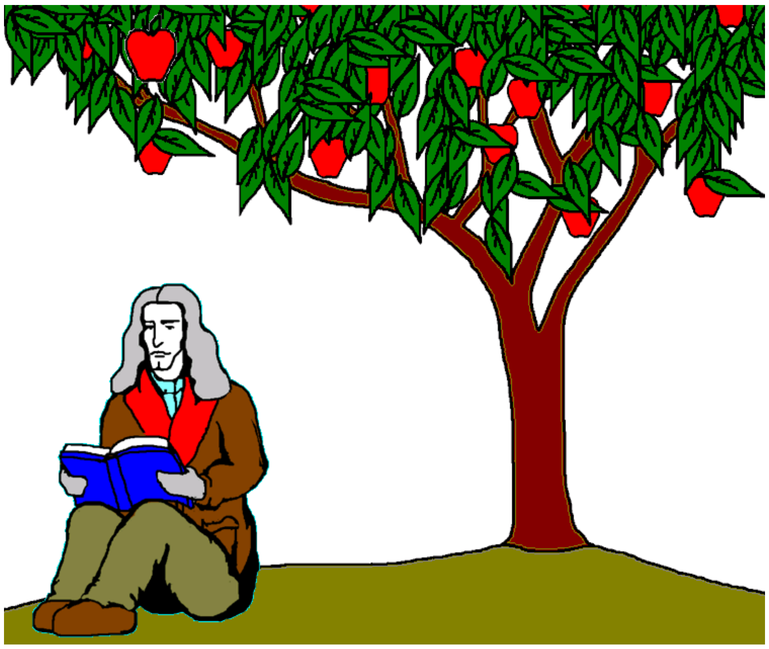
To attach the picture of Newton to the curved polygon:

1. **Click and select the picture of Newton, then hold down the [Shift] key while you click on the curved polygon that represents Newton.**

2. **Select Attach Picture from the Object menu.**

   Notice that the picture of Newton is attached to the curved polygon. The workspace should resemble Figure 4-9.

Figure 4-9
Tree, apple, and Newton



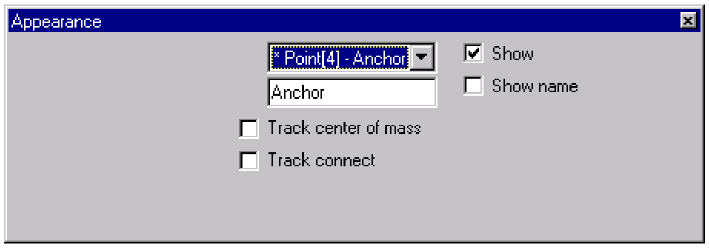
To anchor Newton to the background:

1. **Select the Anchor tool.**
2. **Click in the middle of the picture of Newton.**

To hide the anchor:

1. **Click on the anchor (if it is not already selected).**
2. **Select Appearance from the Window menu (Figure 4-10).**
3. **Uncheck the box labeled Show.**
4. **Close the Appearance window.**

Figure 4-10
Appearance dialog



Show [✓]
Show name [ ]
Track center of mass [ ]
Track connect [ ]

1. **Run the simulation.**

   Notice that the apple falls from the tree and hits Newton.

2. **Click Stop and Reset.**

4.9. Adding Text to a Simulation

This section describes how to add text to a simulation and change its font, font style, and size. The text is made to appear after the apple hits Newton's head, and then blinks on and off.

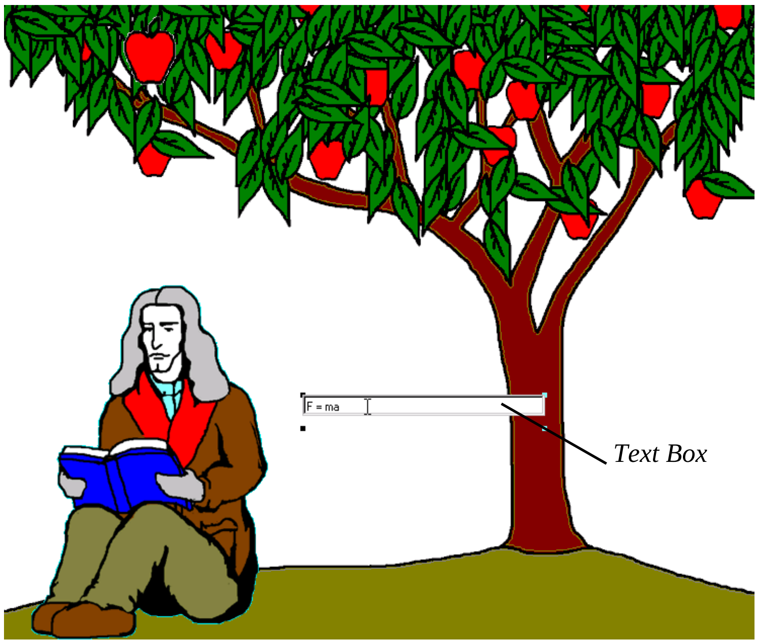
To add text and change its font, size, and color:

1. **Click on the Text tool.**

2. **Click in the white space between Newton and the tree, as shown below.**

   A text box appears (Figure 4-11).

Figure 4-11
Adding text to a simulation



3. **Type F = ma in the text box.**

4. **Select Font... under the Object menu.**

   The Font dialog appears (Figure 4-12).

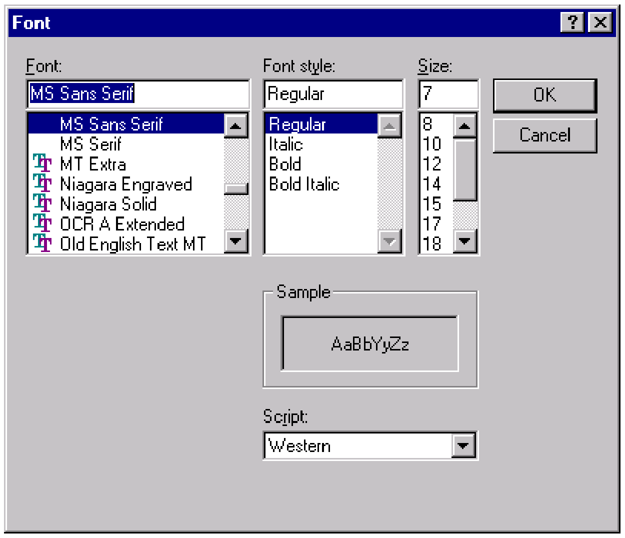


Figure 4-12
Font dialog

Font

Font: MS Sans Serif

Font style: Regular

Size: 7

Sample: AaBbYyZz

Script: Western

5. **Change the Font, Font style, and Size and click [OK].**

   For example, change the Font Style to Bold and the size to 36.

6. **Select Appearance under the Windows menu.**

   The Appearance dialog appears (Figure 4-13).

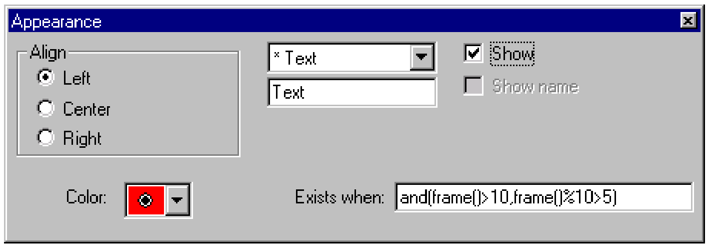


Figure 4-13
Exists when field in the Appearance dialog

Appearance

Align: [✓ Left] Center Right

* Text

Text

[✓ Show] Show name

Color: 🟥

Exists when: and(frame()>10,frame()%10>5)

7. **Change the color of the text.**

To make the text appear after 10 frames and then blink after that:

1. **In the Appearance dialog, enter the following formula in the Exists when box: and(frame()>10, frame()%10>5)**

   This tells the text to appear when:

a) the simulation has run for 10 frames AND
b) the remainder of the number of frames divided by 40 is greater than 20

This controls the frequency and duration of the blinking text.

The symbol "%" is the remainder operator. The formula "X%Y" divides X by Y and returns the remainder of the operation.

2. **Press Enter and close the Appearance window.**

3. **Click Run and observe that the text "F = ma" appears and blinks after the apple falls and bounces off Newton's head.**

   The simulation should resemble Figure 4-14.

4. **Click Stop and Reset.**

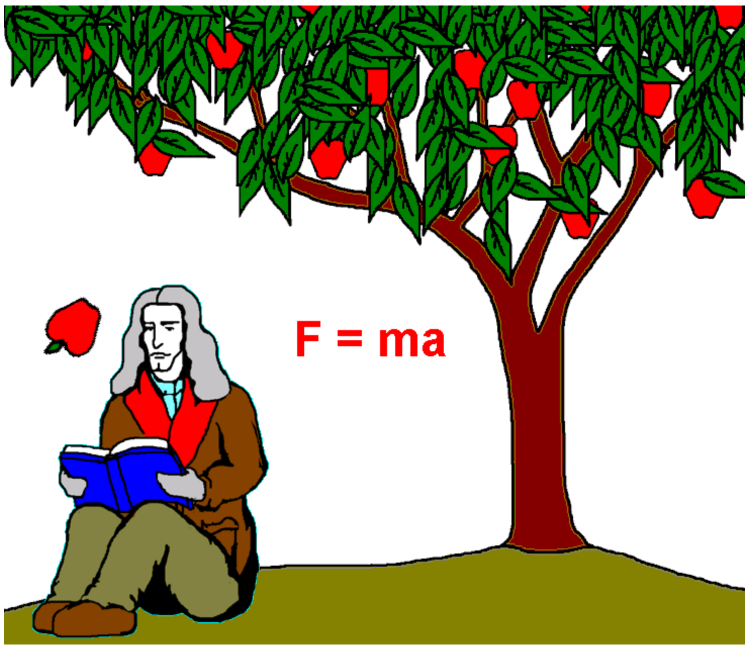


Figure 4-14
Simulation with text showing

4.10. Other Visual Tips

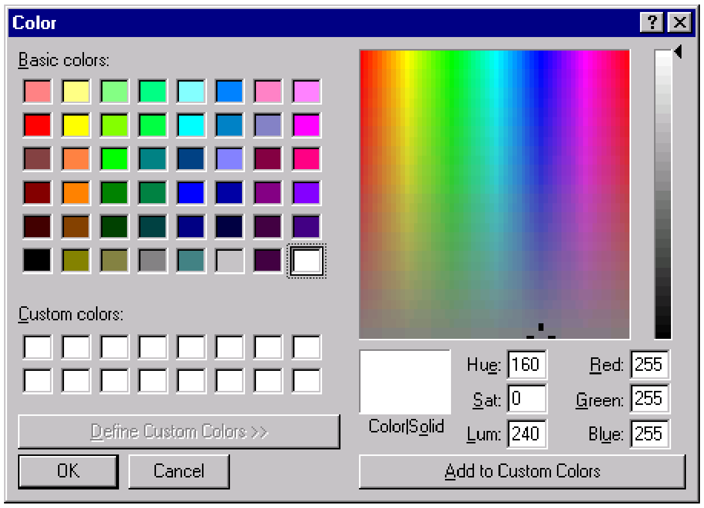
Changing Background Color

The default background color is white. To change the background color:

1. **Select Background Color... from the View menu.**

   The Color dialog appears (Figure 4-15). This is the standard Windows Color dialog box.

Figure 4-15
Color dialog



Hue: 160 Sat: 0 Lum: 240
Red: 255 Green: 255 Blue: 255

2. **Select a color from the Basic colors frame.**
3. **Click [OK].**

   The background color of the workspace is now of the color you have selected.

Using Tracking

You can use Tracking to visually track the path of a body's motion.

To enable Tracking:

1. **Open the Working Model document NewtonAndApple.wm (if it is not already opened).**
2. **Select Tracking under the World menu. From the Tracking submenu, select Every 2 frames.**
3. **Run the simulation.**

   Notice the apple's path as it falls and bounces off Newton's head.

To disable Tracking:

1. **Select Tracking under the World menu. From the Tracking submenu, select Off.**
2. **Run the simulation.**

Changing the Color and Size of Vectors

You can change the size and color of vectors in your simulation.

1. **Open the Working Model document NewtonAndApple.wm (if it is not already opened).**

   The apple has a velocity vector attached to it.

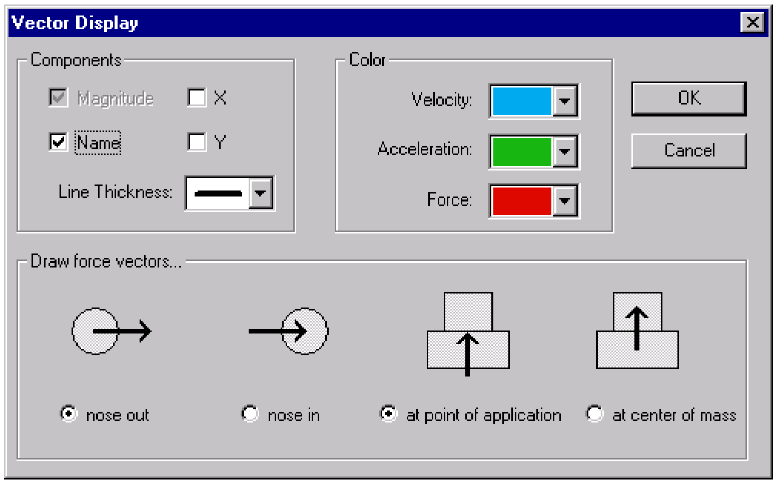
2. **Click Run and note the color and size of the velocity vector.**
3. **Click Stop and Reset.**

To change the size and color of the vector:

1. **Select Vector Display... from the Define menu.**

   The Vector Display... dialog appears (Figure 4-16).

Figure 4-16
Vector Display dialog



2. Change the color of the velocity vector.

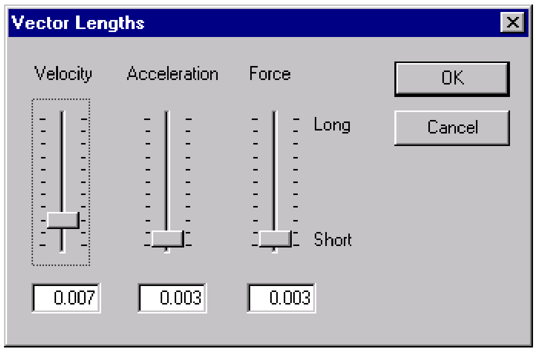
3. Change the line thickness.

4. Click [OK] to close the dialog.

5. Select Vector Lengths... from the Define menu.

The Vector Lengths... dialog appears (Figure 4-17).

Figure 4-17
Vector Length dialog



6. Slide the Velocity slider upward slightly.

7. Click [OK] to close the dialog.

8. **Click Run.**

   Notice that the velocity vector on the apple is now thicker, longer, and of a different color.

Exporting an AVI movie

After you have created the simulation, you can export it to an AVI movie. You can then show the simulation in web pages as a movie.

To export the simulation to a movie:

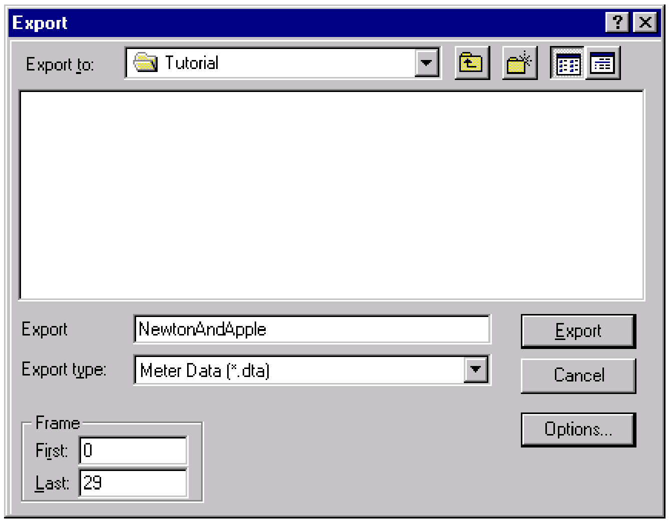
1. **Click Run to start the simulation and click Stop after the apple falls and bounced off Newton and the formula "F = ma" blinks twice.**

   By default, the AVI movie captures the number of frames the simulation has run.

2. **Select Export... under the File menu**

   The Export window appears (Figure 4-18).

Figure 4-18
Export window



Export to: Tutorial
Export: NewtonAndApple
Export type: Meter Data (*.dta)
Frame
First: 0
Last: 29

3. **Select a directory where you want to save the movie.**

4. **Enter the name NewtonAndApple for the movie in the Export field.**

5. **Select Video for Windows (*.avi) in the Export type field.**

6. **Click Export.**

   The status bar at the bottom of the workspace is updated as the movie is being captured.

To watch the movie:

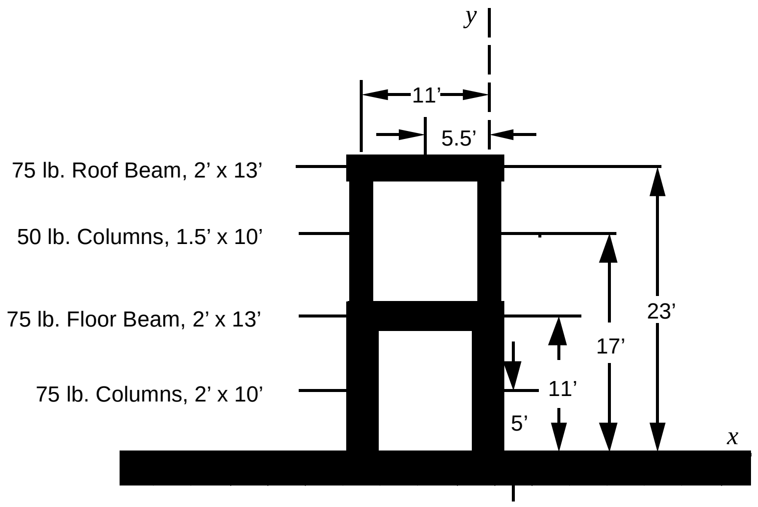
1. **Using Windows Explorer, navigate to the directory where you have saved the movie.**

2. **Double-click on the file NewtonAndApple.avi to play it.**

This opens the default video player on your computer that is associated with the file extension .avi, such as Windows Media Player.

E X E R C I S E 5

An Earthquake Simulation


y
11’
5.5’
75 lb. Roof Beam, 2’ x 13’
50 lb. Columns, 1.5’ x 10’
75 lb. Floor Beam, 2’ x 13’
75 lb. Columns, 2’ x 10’
23’
17’
11’
5’
x

The two story building above is to be tested for its earthquake integrity. Assume that the beams and columns are not attached by moment resisting (earthquake resistant) connections or bracing. If the building is subjected to earthquake forces, qualitatively describe the consequences. Assume the earthquake motion follows the formula:

$$10\cos(7t) + 7\sin(3t)$$

Exercise 5 Concepts:

- Creating actuators
- Modifying the material properties of objects

5.1. Introduction

Structural engineers, particularly those in the western United States, need to test buildings for their integrity against the forces of earthquakes. To test structures, they often build a scale model and place it on a specially designed piece of equipment known as a shake table. A shake table is often actuated by hydraulic cylinders which are capable of mimicking the lateral motions of an earthquake.

In this exercise, you will build a model structure from rectangular bodies. You will create a shake table on which to test the building's integrity. The shake table will be composed of a large rectangle attached to the background by a keyed slot joint. The earthquake motion will be provided by an actuator.

5.2. Setting Up the Workspace

In this exercise, you will use the English unit system of pounds and feet. To verify the current unit system:

1. **Choose Numbers and Units… from the Views menu.**

   The Numbers and Units dialog appears.

2. **Choose English (Pounds) from the Units System pop-up menu if it is not already selected (see Figure 5-1).**

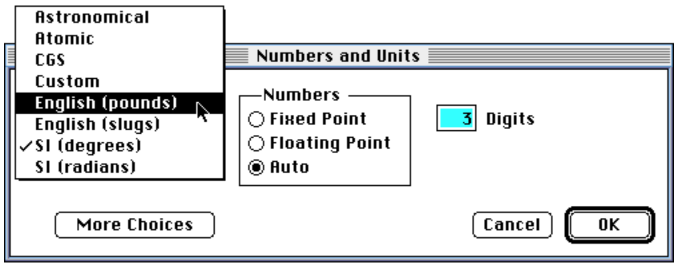


Figure 5-1
Setting the unit system in the Numbers and Units dialog

3. **Click More Choices to expand the Numbers and Units dialog.**

4. **Click the distance unit pop-up menu. Change the unit to feet.**

5. **Click OK**

The Numbers and Units dialog is closed.

To change the simulation accuracy

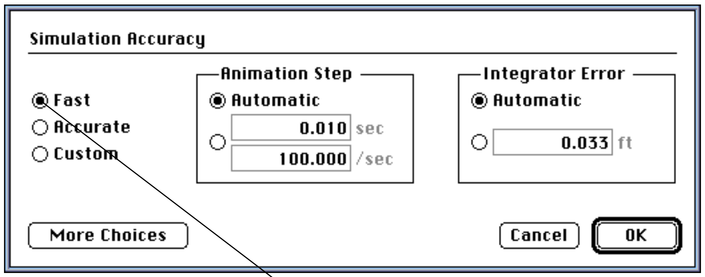
1. **Choose Accuracy… from the World menu.**

   The Simulation Accuracy dialog appears.

2. **Choose Fast in the Simulation Accuracy dialog (see Figure 5-2).**

   Since this exercise requires only a qualitative analysis, we will use the Fast mode for quicker animation.

Figure 5-2
Simulation Accuracy dialog



Simulation Accuracy

(● Fast) ○ Accurate ○ Custom

Animation Step: (● Automatic) ○ 0.010 sec / 100.000 /sec

Integrator Error: (● Automatic) ○ 0.033 ft

3. **Click OK**

To add the x-y axes:

1. **Choose Workspace from the View menu.**

2. **On MacOS systems, choose X,Y Axes from the Workspace submenu. On Windows systems, check the box next to X,Y Axes in the Workspace dialog.**

   The x- and y-axes are displayed.

5.3. Creating the Shake Table

In this exercise, a large shake table is used to simulate the ground motion of an earthquake. The shake table will be constructed of a large rectangle which will slide on a keyed slot joint. The earthquake motion will be simulated using an actuator.

Drawing the Shake Table

The shake table rectangle will be drawn using the Rectangle tool. To draw the shake table rectangle:

1. **Click the Rectangle tool in the Toolbar.**

2. **Click the background and drag out a large rectangle.**

For this exercise the shake table rectangle must be large. Though its exact size is not important, we will use a height of 10' and a width of 70'. These dimensions will be set using the Coordinates bar.

To set the size of the rectangle:

1. **Select the shake table rectangle if it is not already selected.**

2. **Click the Height field of the Coordinates bar and enter 10.**

3. **Tab to the Width field of the Coordinates bar and enter 70.**

   The rectangle is sized accordingly.

Zooming Out

The rectangle will appear large after sizing. To make the rectangle appear smaller and easier to manipulate:

1. **Click the Zoom Out tool in the Toolbar.**

   On MacOS systems, the Zoom Out tool is "hidden" in the Zoom pop-up palette by default. Click and hold on the Zoom In tool to bring the pop-up palette in view (Figure 5-3).

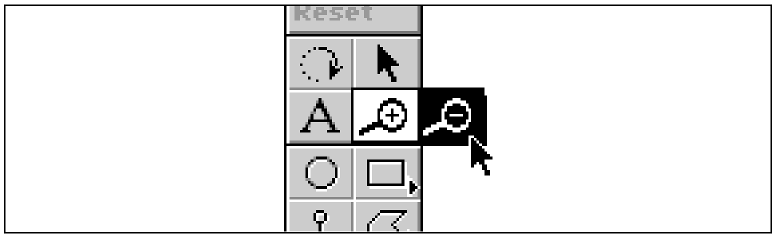


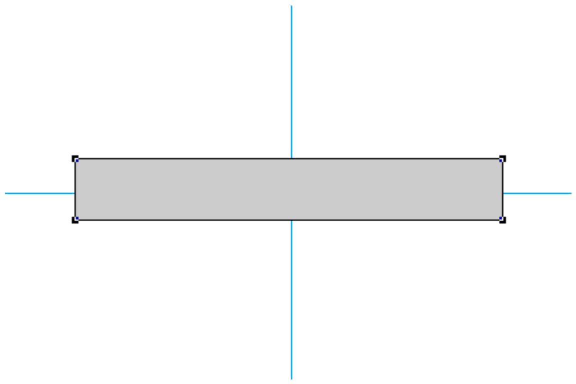
Figure 5-3
Zoom pop-up palette (MacOS only)

2. **Click on or near the rectangle. You may need to click twice before the rectangle fits in the screen.**

   The Zoom Out tool reduces by a factor of two on each mouse click.

   Scroll the window so that the rectangle is in the center of the screen as in Figure 5-4 below.

Figure 5-4
The shake table after zooming out



3. **Click the Arrow tool or press the spacebar to deselect the Zoom Out tool.**

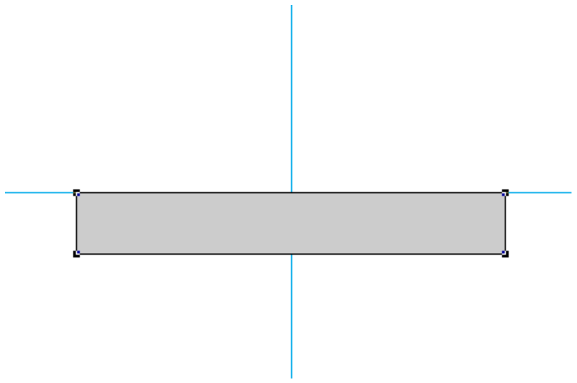
Positioning the Shake Table

For clarity, the shake table will be centered horizontally on the y-axis and positioned just below the x-axis. To position the rectangle:

1. **Select the rectangle if it is not already selected.**
2. **Click the X field of the Coordinates bar and enter 0.**
3. **Tab to the Y field of the Coordinates bar and enter -5.0.**

   Your window should resemble Figure 5-5 below.

Figure 5-5
The shake table positioned



Creating the Shake Table Slot

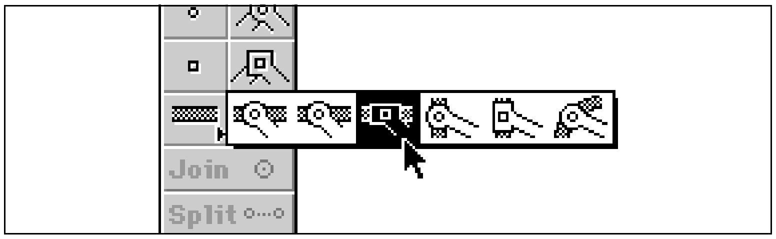
In this exercise, the shake table must only move in the horizontal direction. To provide this constraint, a keyed slot joint will be attached to the shake table rectangle. The position of the slot on the rectangle is not critical.

To create the keyed slot joint:

1. **Choose the Horizontal Keyed Slot tool in the Toolbar.**

   On MacOS systems, this tool is "hidden" in the Slot Joint pop-up palette by default. Click and hold on the Horizontal Pinned Slot tool to bring the pop-up palette in view (Figure 5-6).

Figure 5-6
Slot joint pop-up palette (MacOS only)



2. **Click the shake table rectangle.**

   This tool creates a square point on the object and a slot on the background, and then automatically joins them. Your window should resemble Figure 5-7 below.

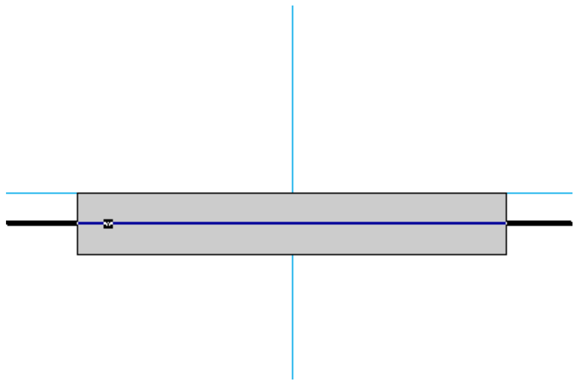


Figure 5-7
The keyed slot connected to the shake table

The Smart Editor will keep joints together during editing. To see this, try dragging the rectangle. The rectangle will stay joined while it is dragged. It will not rotate or move vertically. When you are done dragging the rectangle, return it to the approximate starting position.

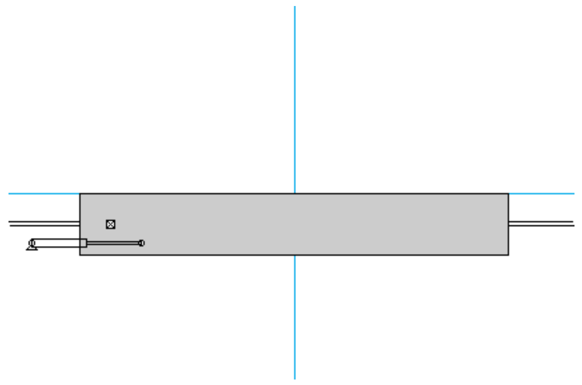
Creating the Actuator

To provide horizontal motion to the shake table, an actuator will be attached to the rectangle and to the background. The Actuator tool will be used to create the actuator. To create the actuator:

1. **Click the Actuator tool in the Toolbar.**
2. **Click the background to the left of the shake table. Move the pointer to the right to extend the actuator. Click again when the pointer is over the rectangle (see Figure 5-8).**

   An actuator appears and joins to both the background and the rectangle. Your window should resemble Figure 5-8.

Figure 5-8
The actuator attached



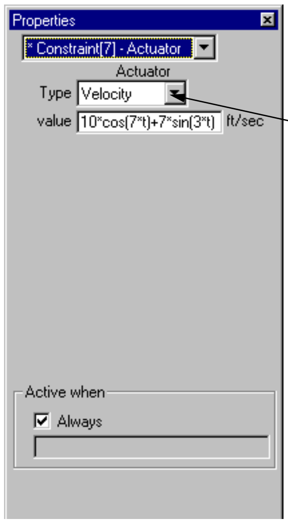
Initializing the Actuator

For this exercise, a Velocity actuator will be used. To set the type and speed of the actuator, the Properties window will be used. To set up the actuator:

1. **Select the Actuator if it is not already selected.**
2. **Choose Properties from the Window menu.**
3. **In the Properties window, select Velocity from the Type pop-up menu (see Figure 5-9).**

   A velocity actuator provides the specified speed regardless of the forces required.

Figure 5-9
Type pop-up menu in the Properties window for an actuator



4. **Click the Value field and enter the following “Earthquake” equation:**

```
10*cos(7*t) + 7*sin(3*t)
```

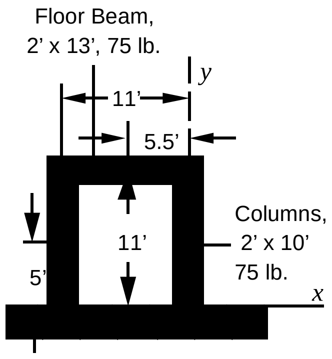
Testing the Shake Table

To test the shake table, run the simulation. The shake table should move left and right on its slot. To run the simulation:

1. **Click the Run button in the Toolbar.**

   The shake table should move left and right. If it does not, the model was not constructed properly.

2. **Click the Reset button.**



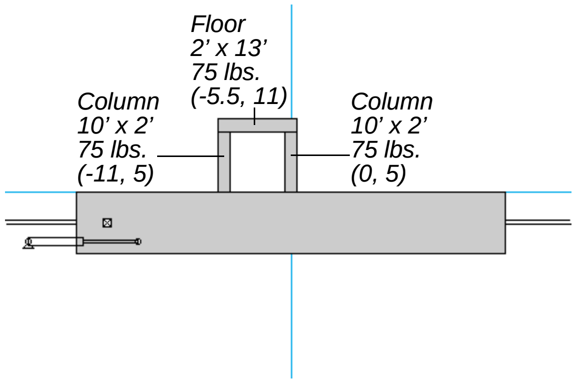
5.4. Creating the First Story

The first story of the building consists of three components: the two columns and the floor beam. The columns weigh 100 lbs. and are 10’ high and 2’ wide. The floor beam weighs 75 lbs. and is 2’ high and 13’ wide. These members will be modeled by rectangular bodies. To simulate simple non-earthquake resistant member connections, the friction and elasticity coefficients will should be set at 1.0 and 0, respectively.

You can follow the step by step instructions or you can create the first story yourself. Refer to Figure 5-10 for the components' data.

Figure 5-10

First story member sizes and positions



Creating the Columns

One column will be created using the Rectangle tool. The Geometry window will be used to set its dimensions, and the Properties window will be used to set its mass. It will then be duplicated to create the second column.

To draw the rectangle:

1. Click the Rectangle Tool in the Toolbar.

2. Click the background and drag out a rectangle.

The dimensions of the column are 10' high and 2' wide. We will position and size the rectangle exactly using the Coordinates bar:

1. Select the rectangle if it is not already selected.

2. Click the X field of the Coordinates bar and enter 0.

3. Tab to the Y field of the Coordinates bar and enter 5.

4. **Tab to the Height field of the Coordinates bar and enter 10.**

5. **Tab to the Width field of the Coordinates bar and enter 2.**

The rectangle is sized and positioned precisely on the shake table.

To set the mass of the column rectangle:

1. **Choose Properties from the Window menu.**

2. **Click the Mass field and enter 75.**

To duplicate the rectangle:

1. **Select the column rectangle if it is not already selected.**

2. **Choose Duplicate from the Edit menu.**

   The Duplicate command can also be selected by pressing Command+D (MacOS) or Control+D (Windows) .

To position the duplicated column:

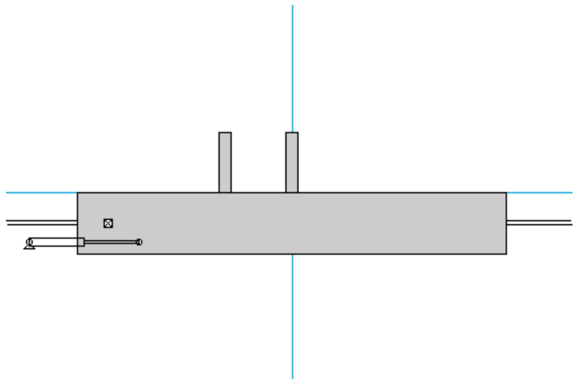
3. **Select the duplicated column.**

4. **Click the X field of the Coordinates bar and enter -11.**

5. **Tab to the Y field of the Coordinates bar and enter 5.**

Both columns are positioned precisely on the shake table (Figure 5-11).

Figure 5-11
First story columns in place



Creating the Floor Beam

The floor beam rectangle will be created with the Rectangle tool. To draw the rectangle:

1. **Click the Rectangle tool in the Toolbar.**
2. **Drag out a rectangle.**

To size and position the rectangle:

1. **Select the floor beam rectangle if it is not already selected.**
2. **Click the X field of the Coordinates bar and enter -5.5.**
3. **Tabl to the Y field of the Coordinates bar and enter 11.**
4. **Tab to the Height field of the Coordinates bar and enter 2.**
5. **Tab to the Width field of the Coordinates bar and enter 13.**

To set the mass of the floor beam:

1. **Choose the Properties from the Window menu.**
2. **Click the Mass field and enter 75.**

Your screen should resemble Figure 5-12 below.

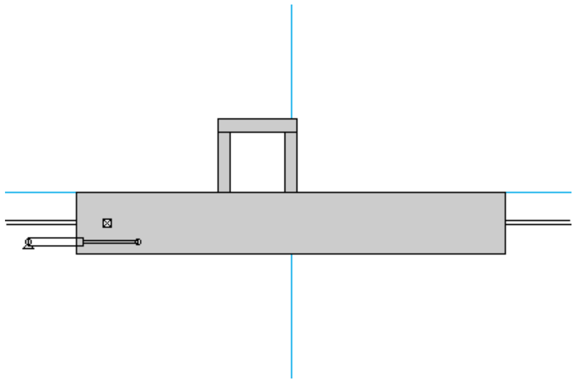


Figure 5-12
The first story completed

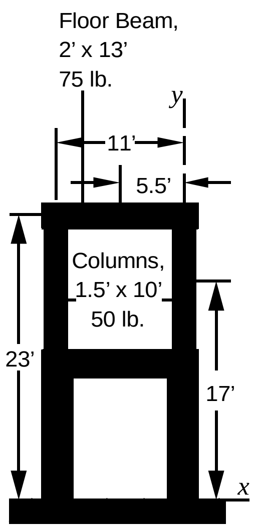
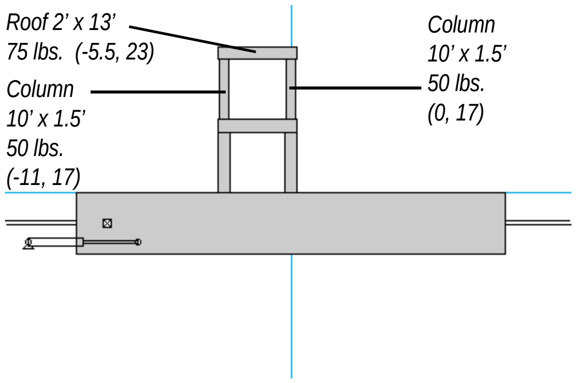


Figure 5-13
Second story member sizes and positions

5.5. Creating the Second Story

The second story of the building in this exercise is similar to the first with the exception that the columns are weigh less and are narrower. One column will be created and then duplicated to create the second column. The roof beam will be created by duplicating the first story floor beam. It will then be positioned on top of the second story columns.

You can follow the step by step instructions or you can create the second story yourself. Refer to Figure 5-13 for the necessary data.



Creating the Columns

To draw one column rectangle:

1. Click the Rectangle tool in the Toolbar.

2. Drag out a rectangle.

To size and position the rectangle:

1. Select the column rectangle if it is not already selected.

2. Click the X field of the Coordinates bar and enter 0.

3. Tab to the Y field of the Coordinates bar and enter 17.

4. Tab to the Height field of the Coordinates bar and enter 10.

5. Tab to the Width field of the Coordinates bar and enter 1.5.

To set the mass of the rectangle:

1. **Choose Properties from the Window menu.**

2. **Click the Mass field and enter the value 50.**

Because the second story columns are identical, the **Duplicate** command will be used to create the next column.

To duplicate the rectangle:

1. **Select the newly created column rectangle if it is not already selected.**

2. **Choose Duplicate from the Edit menu.**

   A second rectangle appears.

Two columns must be placed on opposite ends of the first floor beam. The rectangles will be positioned using the Coordinates bar. To position the second column:

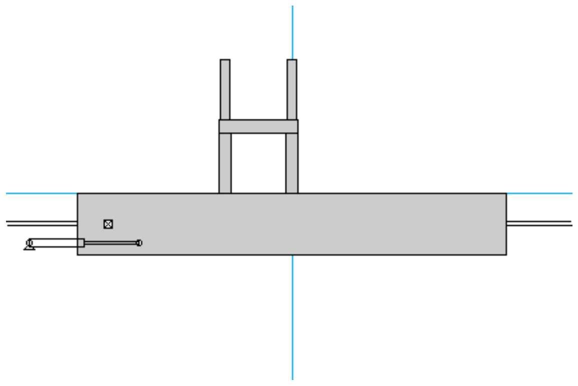
3. **Select the second column.**

4. **Click the X field of the Coordinates bar and enter -11.**

5. **Tab to the Y field of the Coordinates bar and enter 17.**

   The column are now in their proper position (see Figure 5-14).

Figure 5-14
The second story columns in place



Creating the Roof Beam

Since the roof beam rectangle is identical to the first story floor beam, it will be created by duplicating the floor beam. It will then be placed above the second story column rectangles using the Properties window. To duplicate the floor beam:

1. **Select the floor beam.**

2. **Choose Duplicate from the Edit menu.**

   A copy of the floor beam appears.

The roof beam in this exercise is located on the top of the second story columns at the global coordinates x = -5.5 and y = 23. To position the beam:

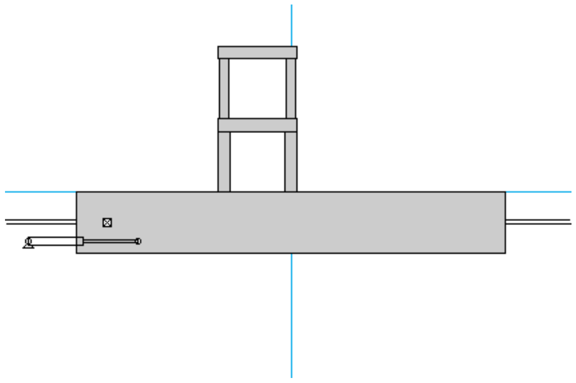
1. **Select the newly duplicated rectangle (roof beam).**

2. **Click the X field of the Coordinates bar and enter -5.5.**

3. **Tab to the Y field of the Coordinates bar and enter 23.**

   The rectangle moves to its proper position (Figure 5-15).

Figure 5-15
All the members in place



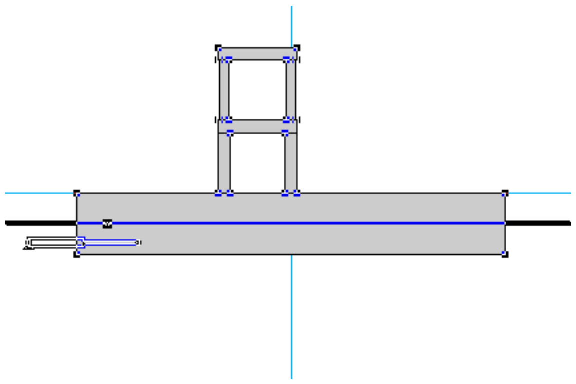
Modifying Elasticity and Friction

The problem states that the building member connections are not earthquake resistant (simple connections). The simple connections will be modeled by setting the member's coefficient of friction to 1 and the elasticity to zero. To modify these properties:

1. **Choose Select All from the Edit menu to select all the rectangles (see Figure 5-16).**

   All the objects display square resize handles at their corners.

Figure 5-16
All the rectangles selected



2. **Choose Properties from the Window menu.**
3. **Click the Static Friction field and enter the value 1.0.**
4. **Click the Kinetic Friction field and enter the value 1.0.**
5. **Click the Elasticity field and enter 0.**

5.6. Decreasing the Integrator Error

Integrator Error represents the error tolerance, or upper bound for numerical errors allowed in the integration process. In this simulation, the Automatic Integrator Error (default) is not sufficiently accurate. The Simulation Accuracy dialog will be used to decrease the error tolerance.

To decrease the Integrator Error:

1. **Choose Accuracy… from the World menu.**

The Simulation Accuracy dialog appears.

2. **Click the Integrator Error field and enter the value 0.05.**

5.7. Running the Simulation

The simulation is ready to run. To run the simulation:

1. **Click the Run button in the Toolbar.**

   The shake table should move left and right. As it does, the building begins to topple over.

When it is done, reset the simulation.

2. **Click the Reset button.**

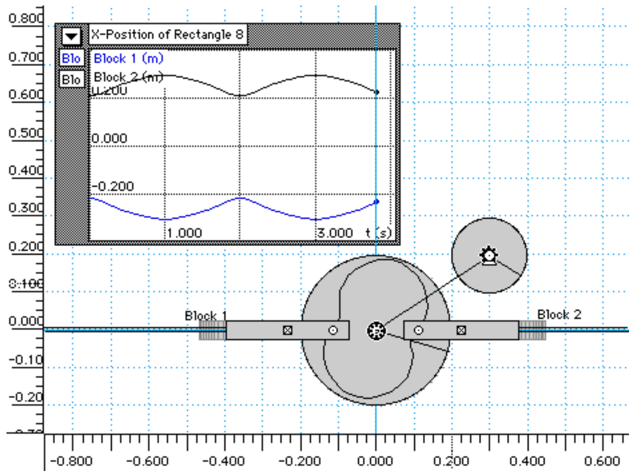
   Now that the simulation has been recorded, play it back at full speed.

3. **Click Run again.**

Try changing the equation in the actuator. Add the rand() function so that the motion of the shake table is more realistic. Also, try modifying the masses of the members.

E X E R C I S E   6

A Belt-Driven Camshaft



In this model, a motor powers a single cam utilizing a belt-drive mechanism. The cam drives two followers in opposing directions. The motor rotates at 600 rpm. You will observe the motion of the followers under the given cam geometry.

Exercise 6 Concepts:

- Creating cams using curved slots
- Using gears to simulate a belt drive mechanism
- Using rulers to draw precise shapes

For this example, you will need the text file that contains the control points necessary to construct the cam geometry. The file (named **Cam Points** on MacOS systems and **CAMPOINT.TXT** on Windows systems) is located in the Tutorial directory under the Samples directory in the

Working Model installation directory, for example, C:\Program Files\Working Model 2D\Samples\Tutorial. If you cannot find the file, you can enter the points by hand (this exercise provides the coordinates).

6.1. Introduction

A cam translates rotational motion to and from linear motion without elaborate mechanism designs. However, a cam design on a drawing board may not intuitively suggest the linear motions derived from its rotation.

In this exercise, you will build a belt-driven cam mechanism that drives two pistons in opposing directions. The motor provides torque to maintain a constant angular velocity. The two pistons are attached to the cam slot grooved on a solid disk. Working Model simulates the belt drive using internal gears, while the curved slot is defined by 18 control points, located at every 20 degrees around its circumference. This exercise will provide you with the geometry of the slot that defines the opposing motions of the pistons.

6.2. Setting Up the Workspace

To make drawing and measuring easier, you will take advantage of some optional tools available in Working Model: namely, **Rulers**, **Grid Lines**, and **X,Y Axes**.

To activate these tools:

1. **Choose Workspace from the View menu.**

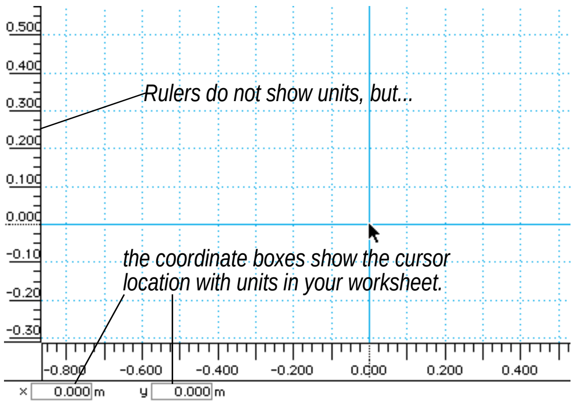
2. **On MacOS systems, activate each of these options from the Workspace submenu. On Windows systems, check the box next to each option in the Workspace dialog.**

Grid Snap should also be activated. This feature allows you to draw and move objects precisely, since every mouse motion is “snapped” to every ruler division.

3. **Pull down the View menu and make sure Grid Snap is active (menu option is checked).**

Observe the rulers and coordinate boxes in your (blank) document. As shown in Figure 6-1, the coordinate boxes show the cursor location with current units.

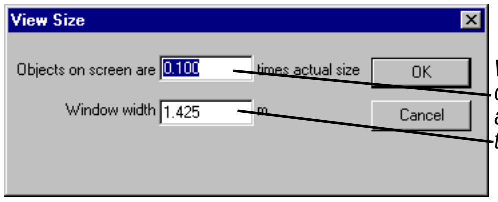
Figure 6-1
Worksheet with workspace options active


0.500
0.400
0.300
0.200
0.100
0.000
-0.10
-0.20
-0.30
-0.800
-0.600
-0.400
-0.200
0.000
0.200
0.400
x 0.000 m
y 0.000 m
Rulers do not show units, but...
the coordinate boxes show the cursor location with units in your worksheet.

To make sure that the worksheet scale is appropriate for the simulation, you will change the view size.

4. **Choose View Size... from the View menu (Figure 6-2), and enter 0.1 in the "Objects on screen are X times actual size" field.**

Figure 6-2
View Size dialog


View Size

Objects on screen are 0.100 times actual size

Window width 1.425 m

OK
Cancel
When you enter a value into one field, Working Model automatically computes the other.

6.3. Creating the Cam

The cam consists of a circular rotating body (disk) with a curved slot grooved on it. The disk is attached to the background with a pin joint.

Drawing the Disk

The disk in this exercise is 40 centimeters in diameter and will be located at the origin. Instead of drawing, locating, and resizing the disk using the Coordinates bar, you will directly draw a disk of the desired size located at the origin.

1. **Click the Circle tool in the Toolbar.**

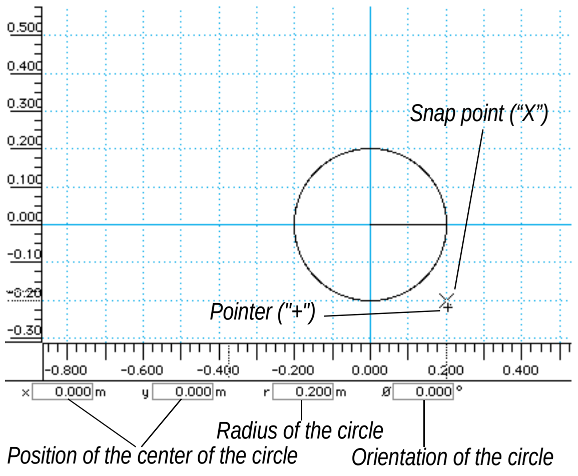
2. **Move the pointer near the point (-0.2, 0.2) in the workspace. (Don't click yet.) Notice that a snap point symbol (an X) appears at the grid intersection.**

   While the snap point symbols appear at each grid division, the dotted lines in the rulers indicating the (x, y) position of the pointer snap precisely to every division of the ruler. The numbers shown in the Coordinates bar also snap to each ruler division.

3. **When the snap point symbol is visible at (-0.2, 0.2), click the mouse button. Move the pointer diagonally down and to the right.**

   Observe that the Coordinate bar tracks the information regarding the circle as shown in Figure 6-3 below.

Figure 6-3
Coordinates bar while creating a circle



4. **When the snap point is visible at (0.2, -0.2), the Coordinates bar shows a radius of 0.2 and the center position as (0.0, 0.0) (as shown in Figure 6-3). Click again to complete the circle.**

You now have a disk with a diameter of precisely 40 centimeters (0.4 meters) located at the origin. If necessary, the Coordinates bar allows you to verify (and modify) the dimension and position of the disk.

Attaching the Disk to the Background

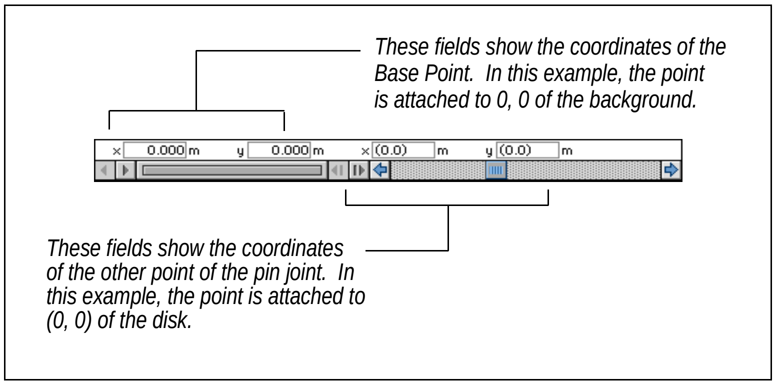
You will use a pin joint to attach the disk to the background at the origin.

1. **Click the Pin Joint tool in the Toolbar.**
2. **Move the pointer to the center of the disk. Look for the snap point at the center.**
3. **Click when the snap point is visible.**
4. **Verify the location using the Coordinates bar.**

   The Coordinates bar should resemble Figure 6-4.

The coordinate values for the disk attachment point are given as (0.0) instead of 0.0 because Working Model automatically generates parametric formulas for snap points to maintain their relative positions as objects are resized. Since 0.0 is not a formula expression, a pair of parentheses is added to denote that the point coordinate was generated as a snap point.

Figure 6-4
Coordinates bar for the pin joint



x 0.000 m y 0.000 m x (0.0) m y (0.0) m

These fields show the coordinates of the Base Point. In this example, the point is attached to 0, 0 of the background.
These fields show the coordinates of the other point of the pin joint. In this example, the point is attached to (0, 0) of the disk.

Drawing the Curved Slot

For this exercise, you will create a closed curved slot that defines the motion of the cam followers. First, you will create an arbitrarily-shaped curved slot on the disk and reshape the slot graphically. Then you will define the precise geometry of the curved slot.

A curved slot consists of a sequence of *control points*. Working Model uses a mathematical algorithm called *cubic interpolation* or *cubic spline* to construct smooth curves between the control points. The spline algorithm is well-known and widely used in the fields of computer aided design and computer graphics.

To create a curved slot:

1. **Choose the Closed Curved Slot tool in the Toolbar.**

   On MacOS systems, the Closed Curved Slot tool is "hidden" in the Slot pop-up palette by default. Click and hold on the Slot tool in the Toolbar to bring the Slot pop-up palette in view (Figure 6-5).

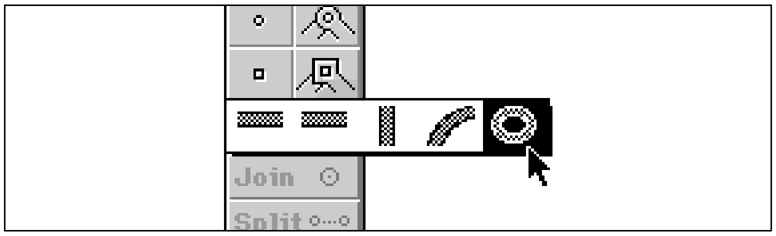


Figure 6-5
Slot Pop-up palette (MacOS only)

   Observe that the pointer changes to a crosshair.

2. **Click once anywhere within the disk. As you move the pointer, observe that a line segment extends from the first point.**

   You have just established the first control point of the curved slot. In addition, by clicking the first point on the disk, you are specifying that the curved slot is attached to the disk.

To create a closed curved slot, you need at least three points.

3. **Click again on the disk in a different place. As you move the pointer, notice that the closed-slot shape changes (Figure 6-6).**

Working Model shows the shape of the closed curve as you move the mouse.

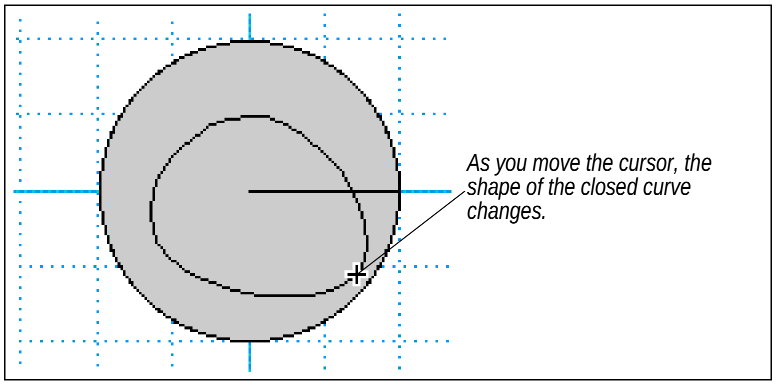

As you move the cursor, the shape of the closed curve changes.

Figure 6-6
Closed curved slot in the making

4. **Click as many times as you like to create more control points.**

   Working Model continuously tracks the point sequence and shapes the slot accordingly. Even if you click outside the disk, Working Model creates a control point as part of the curved slot.

5. **Double-click to create the last control point and finish the curve. Alternatively, press the space bar *after* you create (single-click) the last control point.**

NOTE: Your slot may not look anything like the slots that appear in the exercise. That's okay. In the following steps, you will learn how to change the shape of a curved slot.

Changing the Shape of the Curved Slot

You can modify the shape of a curved slot in two different ways: graphically or numerically. The graphical method provides a simple dragging mechanism for changing the slot shape. The numerical method allows you to position the control points precisely as well as import numerical curve geometry from another application.

First, you will learn how to change the shape graphically. Then, you will use the numerical method to design the cam used in this opposing cam-motion exercise.

Graphical Reshape

You can reshape the curved slot's shape simply by dragging the control points.

1. **Choose Reshape in the Edit menu.**

   You are now in Reshape mode. The control points of the curved slot are shown (see Figure 6-7).

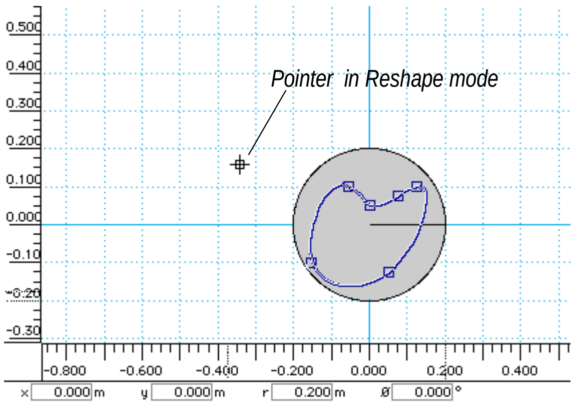


Figure 6-7
Curved slot in reshape mode

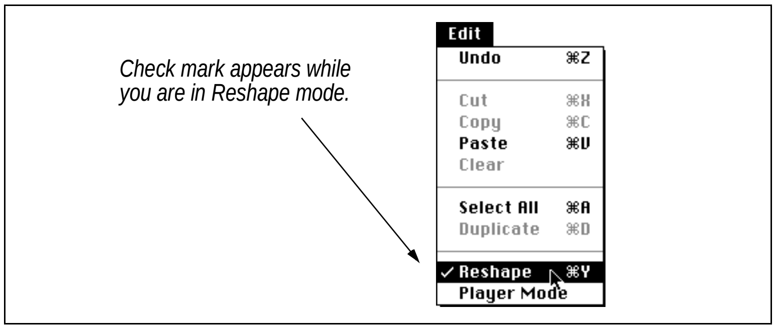
2. **Drag the control point you would like to move.**

   The slot shape changes accordingly.

You can add or delete control points as well. In either case, make sure you are in **Reshape** mode; the pointer should be shaped as a boxed cross-hair. You can also check the **Edit** menu as shown in Figure 6-8.

- To add a control point, click on the slot where you want the new control point to be added.
- To delete a control point, click the point you would like to remove, and press the Delete or Backspace key.

Figure 6-8
Reshape mode active



Numerical Reshape

To construct a cam with precise geometry, you need to enter the coordinates of the control points numerically rather than graphically. In this section, a table of control points is provided for you to enter into Working Model.

First, you will modify a single control point numerically.

1. **Exit Reshape mode.**

   You can exit Reshape mode by selecting the Arrow tool in the Toolbar or by deselecting Reshape in the Edit menu.

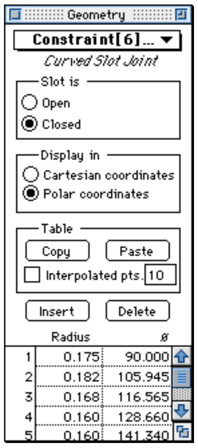
2. **Select the closed curved slot by clicking it.**

   The slot appears highlighted.

3. **Choose Geometry in the Window menu.**

   The Geometry window appears (Figure 6-9). The bottom part of the window shows the coordinates of the control points. It may be necessary to resize the window to see all of the coordinates.

Figure 6-9
Geometry window for a curved slot


Geometry

Constraint[6]...

Curved Slot Joint

Slot is: Open [✓ Closed]

Display in: Cartesian coordinates [✓ Polar coordinates]

Table: Copy | Paste

☐ Interpolated pts. 10

Insert | Delete

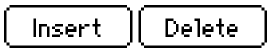
	Radius	ø
1	0.175	90.000
2	0.182	105.945
3	0.168	116.565
4	0.160	128.660
5	0.160	141.340

The control points of a closed curved slot are shown in polar coordinates by default. For an open curved slot, the control points are shown in Cartesian (rectangular) coordinates by default. You can switch between polar and Cartesian coordinate displays by clicking the appropriate button in the Geometry window.

The origin of the control point coordinates is initially located at the frame of reference (FOR) of the object to which the slot is attached—in our example, the center of the disk. If you move the curved slot, the position of its FOR will change accordingly. Reshaping a curved slot does not affect its FOR.

4. **Select one of the coordinate pairs and type in different values to see how the change affects the shape of the curve.**

   Working Model highlights the control point on the slot when you select its coordinates in the Geometry window.



5. **Select one of the coordinates, and click the Delete or Insert button to delete or insert a control point.**

   The Insert button "clones" the selected control point. Enter new coordinate values to make the new point distinct from the original.

To create the precise geometry for your model, the Tutorial folder/directory contains a text file titled **CAMPOINT.TXT**. The file contains the polar coordinates of the 18 control points necessary to generate the shape of the cam you use in this exercise.

6. **Open the control points file *CAMPOINT.TXT* located in the Tutorial folder/directory in the Working Model installation folder/directory.**

   You can use a text editor such as Notepad (on Windows systems) or SimpleText (on MacOS systems), a spreadsheet program, or a word processing program to open the text file.

If you do not have access to any of the programs mentioned above or do not know how to use them, you can type the coordinates directly into Working Model. Shown below is the content of the text file:

Radius	θ
0.100	0.000
0.120	20.000
0.150	40.000
0.170	60.000
0.185	80.000
0.185	100.000
0.170	120.000
0.150	140.000
0.120	160.000
0.100	180.000
0.120	200.000
0.150	220.000
0.170	240.000
0.185	260.000
0.185	280.000
0.170	300.000
0.150	320.000
0.120	340.000

To manually enter the values above, click the Insert button in the Geometry window as many times as necessary to create 18 control points, and then type in these values. Use the Enter or Return key to move from one table cell to the next. *Make sure that the coordinates shown in the Geometry window are polar coordinates.*

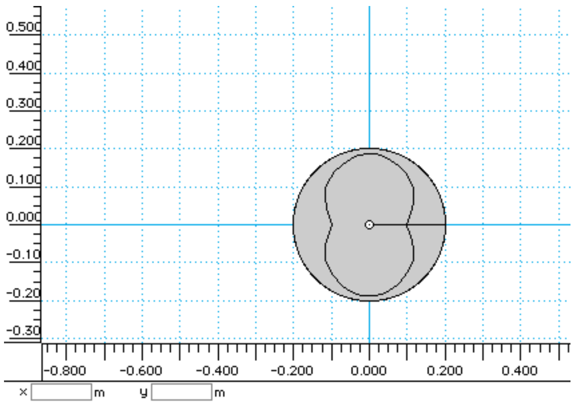
7. **To enter the control points from the file, select the control points in your text editor and use the Copy function of the editor to store the numbers on the Clipboard.**

Paste

8. **Click the Paste button in the Geometry window.**

The coordinates of the control points are pasted into the table regardless of how many control points were present previously. Observe that the curved slot is immediately reshaped to reflect the new coordinates (Figure 6-10).

Figure 6-10
Numerically reshaped cam



Now that you have created the cam, you will create the cam followers and attach them to the cam.

6.4. Creating Cam Followers

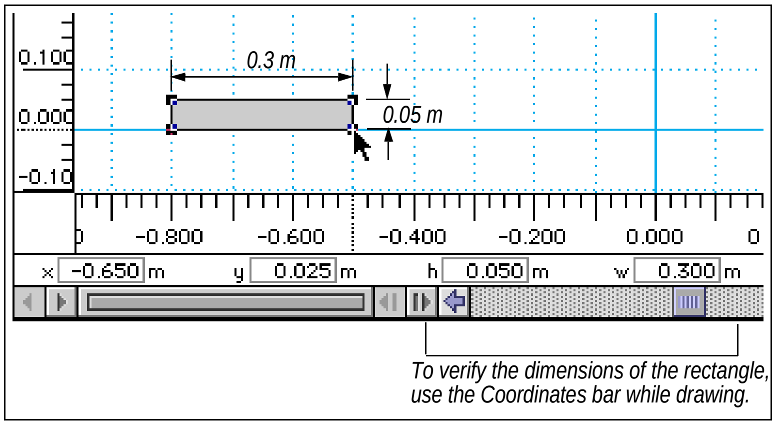
In this exercise, you will create the cam followers as rectangles. Their motion will be restricted to a single degree of freedom by a keyed slot, and their horizontal motion will be defined by the cam.

1. **Click the Rectangle tool in the Toolbar.**
2. **Using the grid lines and Coordinates bar as your guide, create a rectangle of the dimensions shown in Figure 6-11.**

   Use the Coordinates bar to verify and modify the shape (if necessary).

Figure 6-11

Cam follower dimensions

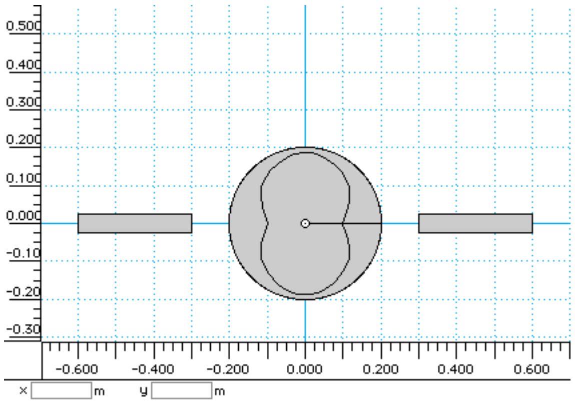


3. **Select the cam follower and choose Duplicate in the Edit menu to create another cam follower of the same dimensions.**

4. **Drag the cam followers in the workspace so that they are on either side of the cam as shown in Figure 6-12.**

Figure 6-12

Preliminary positioning of the cam followers.

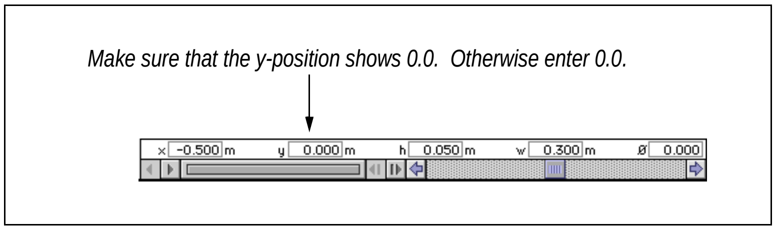


You will now attach the cam followers to horizontal keyed slots in order to restrict their motion. Before the attachment, you will make sure the cam followers are aligned to the x-axis of the workspace.

5. **Click on either cam follower and examine the Coordinates bar (Figure 6-13). Verify that the y-position of each follower is 0.0.**

   For now, their x-position does not need to be precise .

Figure 6-13
Coordinates bar for one of the cam followers


Make sure that the y-position shows 0.0. Otherwise enter 0.0.
x -0.500 m
y 0.000 m
h 0.050 m
w 0.300 m
ø 0.000

6. Choose the Horizontal Keyed Slot tool and click one of the cam followers.

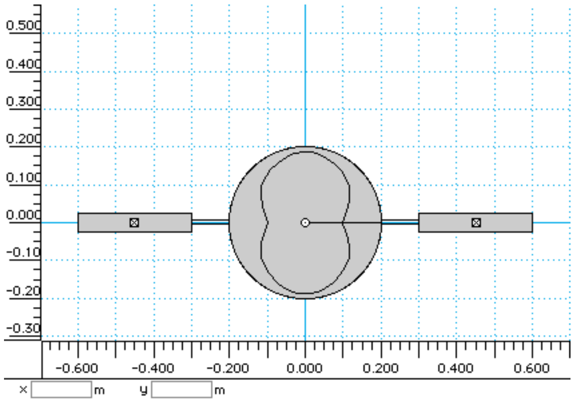
On MacOS systems, the Horizontal Keyed Slot tool may be "hidden" in the Slot Joint pop-up palette. Click and hold on the displayed Slot Joint tool to bring the pop-up palette in view.

Since you are not interested in torques that may be applied at the joint, you do not have to worry about where in the rectangle the keyed slot joint should be located.

7. Repeat step 6 for the other cam follower.

The simulation should resemble Figure 6-14.

Figure 6-14
Two cam followers with keyed slot joints



Creating Attachment Points on the Cam Followers

First, you will use the Point tool to attach two pins to the cam followers.

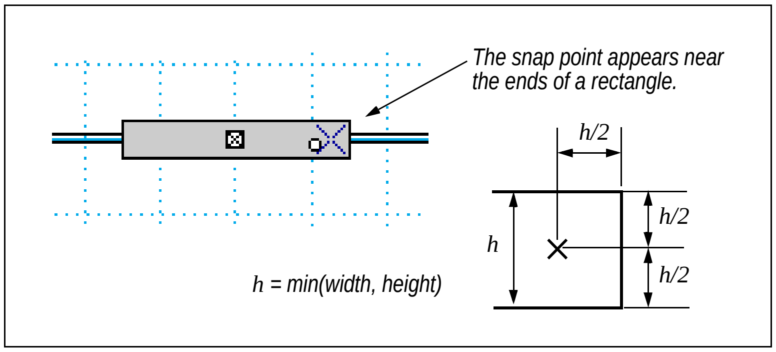
1. **Double-click the Point tool in the Toolbar.**

   Double-clicking on the tool allows you to use the tool repeatedly without going back to the tool palette each time. You will use the tool twice in a row.

2. **Place the mouse pointer near the right end of the left cam follower. Find the snap point near the right end (but not at the edge).**

   Figure 6-15 shows the snap point.

Figure 6-15
Snap point near the right end of the left cam follower

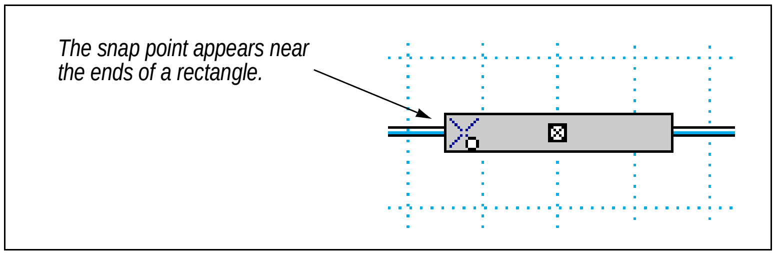


3. **Click when the snap point is visible.**

   The point element is positioned at the snap point.

4. **Repeat the process to attach a point element at the left end of the right cam follower (Figure 6-16).**

Figure 6-16
Snap point near the left end of the right cam follower



Each point needs to be attached to the slot on the cam. However, only one point can be attached to each slot in Working Model. In order to accommodate two point elements, we will create another slot element by duplicating the existing one.

1. **Click the Arrow tool in the Toolbar or press the spacebar to deselect the Point tool.**

2. **Select the curved slot currently attached to the disk.**

3. **Select Duplicate in the Edit menu.**

   An identically shaped curved slot is created at a slightly offset position (Figure 6-17).

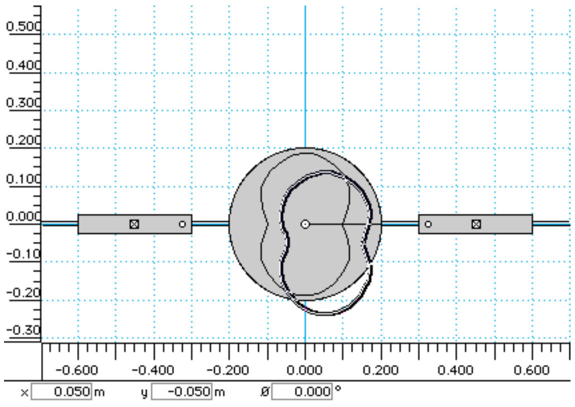


Figure 6-17
Duplicating a curved slot

Before we align the two slots, we will assign names to individual slots.

4. **Select the *original* curved slot and choose Appearance from the Window menu.**

   The Appearance window appears.

5. **Enter "Left Follower Slot" in the name field of the Appearance window.**

   The name indicates that the slot is meant for the left cam follower.

6. **Select the duplicated curved slot and enter "Right Follower Slot" in the name field of the Appearance window.**

FOR of a Curved Slot

In order to align the two curved slots, you have to know how Working Model keeps track of the position of slots.

Each slot has a frame of reference (FOR) that represents the location of the slot. The Coordinates bar for a curved slot shows where the FOR is located (Figure 6-18). The coordinates are measured with respect to the center of the body to which the slot is attached.

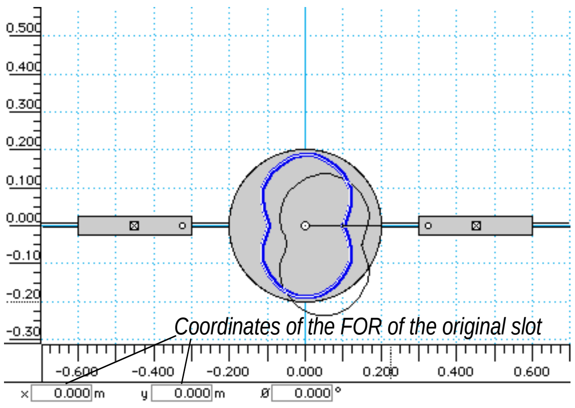


Figure 6-18
Coordinates bar displaying the frame of reference of a curved slot

To bring the duplicated slot directly over the original slot:

1. **Verify the FOR coordinates for the original curved slot (named Left Follower Slot).**

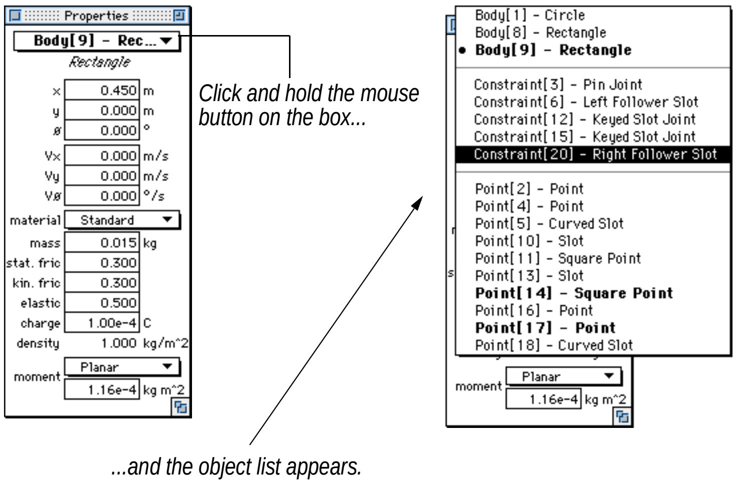
   The coordinates should be (0, 0).

2. **From the selection pop-up menu in the Properties window, select the constraint named Right Follower Slot (Figure 6-19).**

NOTE: The number assignments shown in Figure 6-19 may be different from those in your model. In addition, on Windows systems, the object names are truncated in the selection pop-up display.

Figure 6-19

Selection pop-up in the Properties window.



```
Body[9] - Rec...
Rectangle
x           0.450    m
y           0.000    m
θ           0.000    °
Vx          0.000    m/s
Vy          0.000    m/s
Vθ          0.000    °/s
material    Standard
mass        0.015    kg
stat. fric  0.300
kin. fric   0.300
elastic     0.500
charge      1.00e-4  C
density     1.000    kg/m^2
moment      Planar
            1.16e-4  kg m^2
```

3. **Enter 0 in the x- and y-fields of the Coordinates bar to align the key point location with the Left Follower Slot.**

The two slots are now positioned at the same location.

Attaching the Cam Followers to the Slots

To attach each cam follower to respective slots:

1. **Using the Properties window selection pop-up menu, choose the Left Follower Slot.**

2. **Shift-select the point element attached to the left cam follower.**

3. **Click the Join button in the Toolbar**

   The left cam follower is now pinned to the curved slot.

4. **Choose the Right Follower Slot from the selection pop-up.**

5. **Shift-select the point element located on the right cam follower.**

6. **Click the Join button in the Toolbar.**

Both followers are now attached to the cam disk.

Verifying Your Model

To verify the construction of your model, rotate the cam disk and make sure the followers indeed follow the cam rotation.

1. **Click the cam disk away from its center and the slots, hold down the mouse button, and drag slowly as if you were trying to rotate the disk.**

   The cam followers move as the cam rotates (see Figure 6-20).

   If you selected slots or the motor by mistake and wound up dragging them, simply choose Undo from the Edit menu and start over.

If the rectangles do not “follow” the cam, they may not have been attached properly to the curved slots. Go back to the appropriate sections in this tutorial and review your steps.

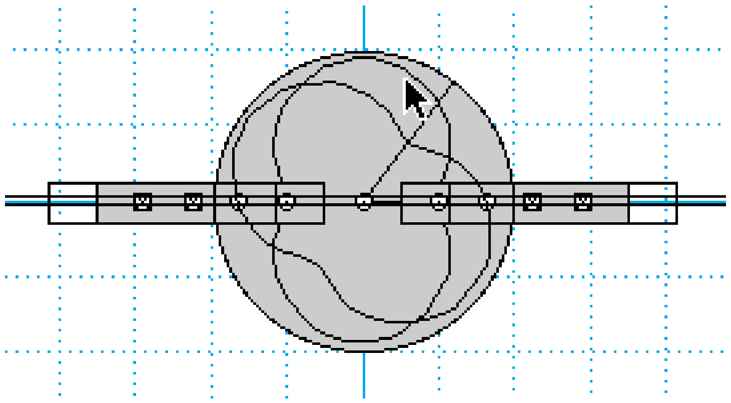


Figure 6-20
Dragging the cam disk

To restore the initial conditions:

1. **Select the circle.**
2. **In the Coordinates bar, enter 0 in the field labeled “ø”.**

   The rotation angle of the disk is reset to 0, and the cam followers are repositioned accordingly.

6.5. Constructing the Drive Mechanism

You will now construct a motor and drive disk and then attach them to the cam to simulate a belt drive mechanism.

Creating the Drive Disk

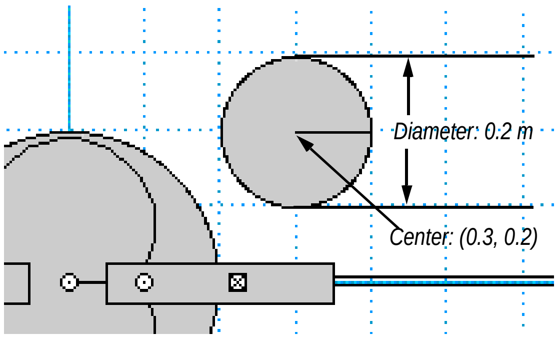
You will use the **Circle** tool to construct a drive disk.

1. **Click the Circle tool in the Toolbar.**

2. **As you constructed the cam disk, create a circle according to the specifications shown in Figure 6-21.**

   The Coordinates bar should indicate the position of the circle as (0.3, 0.2).

Figure 6-21
Drive disk specifications



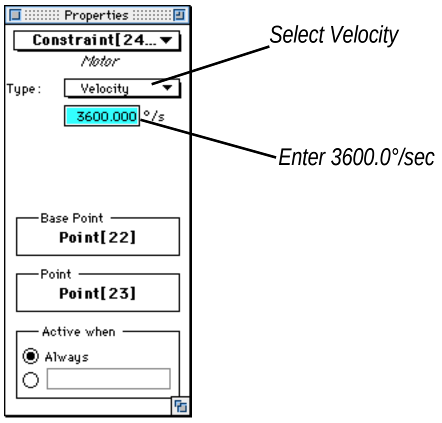
3. **Click the Motor tool in the Toolbar.**

4. **Click the center snap point of the drive disk to attach the motor.**

5. **In the Properties window of the motor (Figure 6-22), select the type Velocity, and enter 3600.0 degrees per second (which equals the specified velocity of 600 rpm).**

   Alternatively, you can set the units of angular velocity to be measured in rpm using the Numbers and Units dialog. See the Working Model User's Manual *for details.*

Figure 6-22
Properties window for a motor



Connecting the Drive Motor to the Cam

As was discussed in the beginning of the exercise, you will use a belt-drive to attach the drive motor to the cam disk. In Working Model, you can use the gear tool to simulate belt-drive mechanisms.[1]

By default, Working Model automatically computes the gear ratio for a pair of circular objects based on their radii.

Interested readers should consult **Chapter 4, “Constraints”** in the *Working Model User’s Manual* for detailed information about the gear simulation principles.

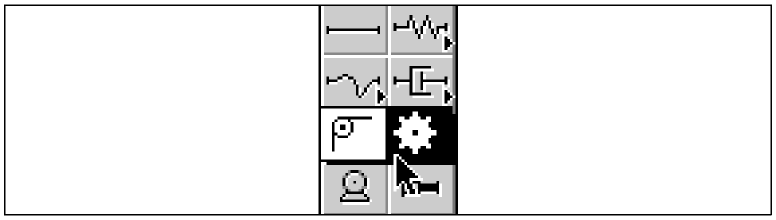
In order to connect the drive disk to the cam disk:

1. **Choose the Gear tool in the Toolbar.**

   On MacOS systems, the Gear tool is "hidden" in the Pulley/Gear pop-up palette by default. Click and hold on the Pulley tool to bring the pop-up palette in view (Figure 6-23).

[1] Since the gear tool in Working Model simulates the dynamics between two bodies, you will not be able to control properties such as the weight or tension of the belt.

Figure 6-23
Pulley/gear pop-up palette (MacOS only)



2. **Click anywhere on the cam disk.**

   A gear icon is automatically aligned to the center of the cam disk. The second gear icon follows the point as you move it.

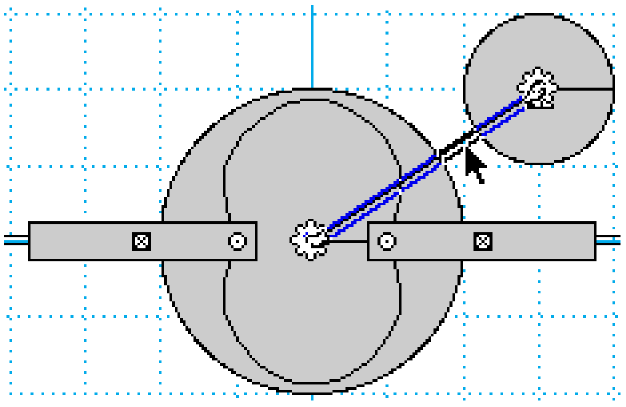
3. **Move the pointer to the drive disk and click the mouse button again.**

   Again, the second gear icon is automatically aligned to the center of the drive disk. Your simulation should resemble Figure 6-24.

At this point, the two disks will behave as if they were spur gears; *i.e.*, the two disks will rotate in opposite directions when the simulation is run. To simulate a belt-drive mechanism:

4. **Select the gear constraint by clicking on the “rod” connecting the two gear icons and open the Properties window if it is not already open.**

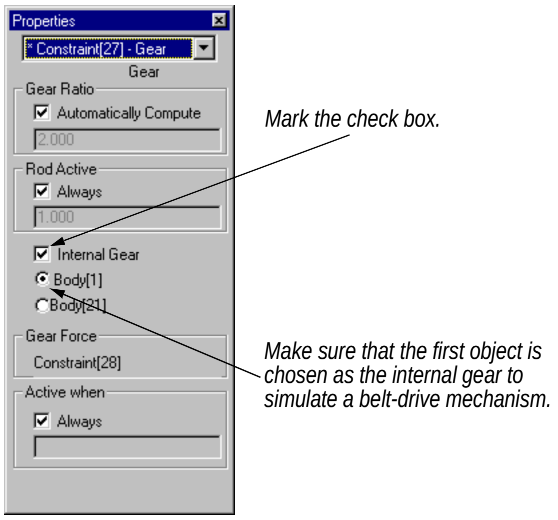
Figure 6-24
Selecting the gear constraint



5. **Check the Internal Gear checkbox. Make sure that the first object you selected as a gear (the cam disk, if you followed the steps above) is chosen as the internal gear (see Figure 6-25).**

   The gear icon on the cam disk turns into an internal gear icon with teeth facing inward.

Figure 6-25
Properties window for gears



Properties
* Constraint[27] - Gear
Gear
Gear Ratio
[✓] Automatically Compute
2.000
Rod Active
[✓] Always
1.000
[✓] Internal Gear
[● Body[1]]
○ Body[21]
Gear Force
Constraint[28]
Active when
[✓] Always

Mark the check box.

Make sure that the first object is chosen as the internal gear to simulate a belt-drive mechanism.

Now your model is complete. Before you click the Run button, however, please read the following section.

6.6. Running the Simulation

As it stands, the motor is supposed to operate at a relatively high speed (600 rpm). The default animation time step is too large for this simulation, making the animation frames appear discontinuous. You will modify a performance parameter of Working Model to alleviate the problem.

Setting Animation Step

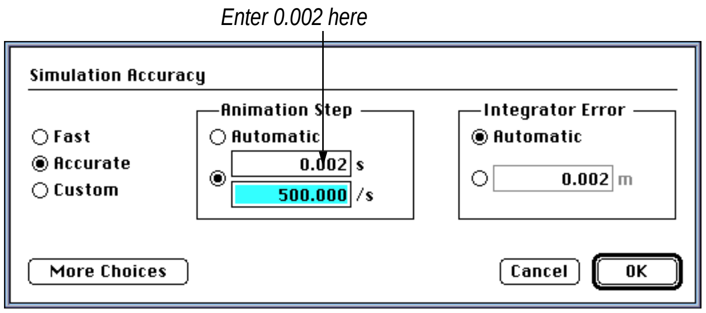
You will adjust the rate at which Working Model refreshes its animation frames.

1. **Choose Accuracy... from the World menu.**

   The Simulation Accuracy dialog box appears.

2. **In the frame labeled Animation Step, click on the radio button located to the left of the two text boxes (under Automatic). Enter 0.002 in the field labeled seconds (Figure 6-26).**

   Each animation frame now represents 0.002 seconds of the simulation.



Simulation Accuracy

○ Fast
● Accurate
○ Custom

Animation Step
○ Automatic
● 0.002 s
500.000 /s

Integrator Error
● Automatic
○ 0.002 m

Figure 6-26
Simulation Accuracy dialog

You can display the simulation time by creating a time meter to observe how the real time corresponds to the state of your mechanism. To create the time meter:

3. **Choose Time from the Measure menu.**

   A digital meter showing the elapsed time appears.

Interested readers should consult **Appendix A, “Technical Information”** in the *Working Model User’s Manual* for detailed discussions on the numerical method employed by Working Model.

Now you are ready to run your simulation.

Starting the Simulation

To start your simulation, as in previous exercises:

1. **Click the Run button in the Toolbar.**

Alternatively, you can press Command+R (MacOS), or Ctrl+R (Windows).

Observe the motion of the cam followers as the motor drives the whole mechanism.

Modifying the Simulation

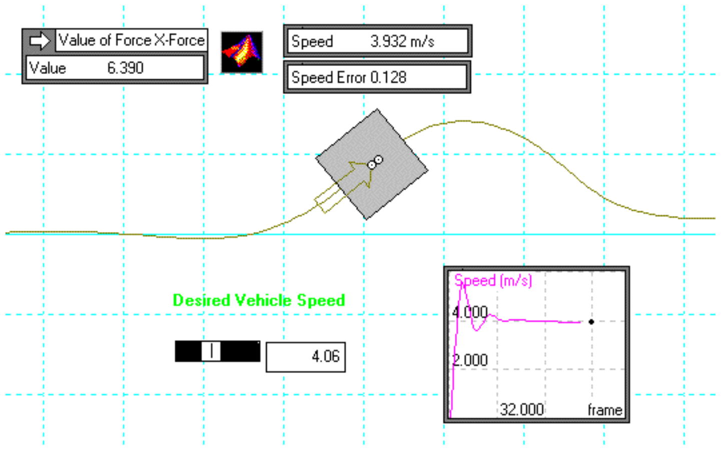
Depending on your application, you might want to:

- measure the velocity and position of the cam followers,
- change the speed of the motor,
- modify the shape of the cam, and/or
- change the animation step size.

By now, you have probably learned enough about Working Model to perform the above tasks. We encourage you to consult the *Working Model User's Manual* for detailed discussions on how Working Model treats and simulates your models.

E X E R C I S E   7

Cruise Control using MATLAB



In this model, a control system monitors the speed of a vehicle and adjusts its throttle. The above picture represents a vehicle moving over a hill. Although the vehicle experiences a gravitational pull on the inclined track, the control system attempts to keep the speed of the vehicle constant as specified by the control slider.

- Since this exercise uses a control system implemented in MATLAB, you must have MATLAB version 4.2 or later which supports Dynamic Data Exchange (DDE) for Windows[1].
- The MacOS version of Working Model supports the Apple® Events Table Suite, which is an external application interface protocol.

[1]. At press time, MATLAB 4.2 does not fully support DDE for Windows NT 3.51.

- This exercise is designed for use with DDE and Windows 95. If you are a MacOS user, however, you can still follow along with this exercise up to but not including the section “Implementing the Control System” on page 7–13. The exercise still contains useful ideas and tips for implementing Working Model simulations.

You can also implement a similar control system on MacOS or Windows systems using a spreadsheet program, such as Microsoft Excel, although this methodology is not covered in the exercise. The only difference from using MATLAB is that you would be implementing a control function using spreadsheet cells instead of variables.

Exercise 7 Concepts:

- Simulating a keyed curved slot using two identical pinned curved slot joints
- Using external application interface to exchange data with MATLAB
- Implementing a well-known control system

7.1. Introduction

Modern automobiles often feature a cruise control system. The system monitors the actual speed of the vehicle and computes the appropriate throttle setting in order to maintain the constant speed specified by the driver. The speed is maintained regardless of wind resistance, gravitational pull, or curvature of the road.

In our exercise, the vehicle is modeled as a rectangular body sliding on a curved slot under the action of a force. The curved slot models a “hilly” road going up and down. The force models the propulsion generated by the vehicle’s engine and is controlled by its cruise control system.

This exercise covers many advanced features implemented in Working Model. You will perform the following tasks in order:

- set up the workspace and accuracy for the simulation
- create a curved track and the vehicle
- attach the vehicle to the track and apply a force on the vehicle
- write the control system function in MATLAB
- set up Control/Meter objects in Working Model

- establish the specific links between Working Model Control/Meter objects and MATLAB variables
- run the simulation

7.2. Setting Up the Workspace

Setting the Unit System

You will use the SI unit system in this example. To verify the current unit system:

1. **Choose Numbers and Units... from the View menu.**
2. **Select SI (degrees) from the Unit System selection pop-up menu if it is not already selected.**
3. **Click OK.**

Setting an Integration Parameter

Since the data exchange with MATLAB takes place at the animation frame rate, the control system running in MATLAB is only able to monitor the speed of the vehicle at that rate.

In order for our control system to be effective, however, the default frame rate of Working Model must be increased.

Therefore, to synchronize Working Model with MATLAB, you need to adjust the integration time step parameter used by Working Model's simulation.

To increase this frame rate:

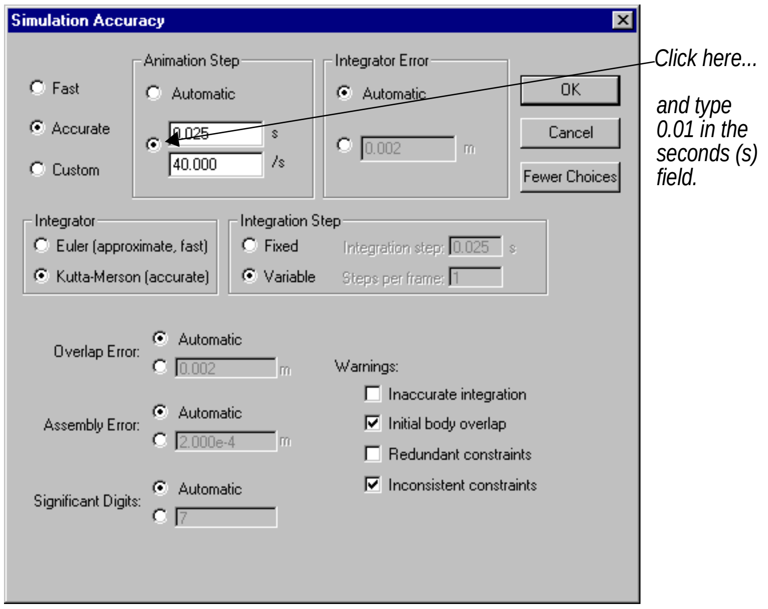
4. **Choose Accuracy... from the World menu.**

   The Simulation Accuracy dialog appears.

5. **Click More Choices.**

   The dialog box expands as shown in Figure 7-1.

Figure 7-1
Simulation Accuracy dialog (Expanded)



( ) Fast
(●) Accurate
( ) Custom

Animation Step
( ) Automatic
(●) 0.025 s
40.000 /s

Integrator Error
(●) Automatic
( ) 0.002 m

Integrator
( ) Euler (approximate, fast)
(●) Kutta-Merson (accurate)

Integration Step
( ) Fixed
(●) Variable
Integration step: 0.025 s
Steps per frame: 1

Overlap Error: (●) Automatic ( ) 0.002 m
Assembly Error: (●) Automatic ( ) 2.000e-4 m
Significant Digits: (●) Automatic ( ) 7

Warnings:
☐ Inaccurate integration
☑ Initial body overlap
☐ Redundant constraints
☑ Inconsistent constraints

By default, the box titled Animation Step is set to Automatic. The numbers below it show the current frame rate.

You will change the frame rate to 0.01s, or 100Hz. By making this change, you are choosing to send measurements data to MATLAB 100 times per (simulated) second.

To change the time step size:

6. **Click the radio button by the frame rate numbers and enter 0.01 in the top field or enter 100 in the bottom field (see Figure 7-1).**

In order to simplify Working Model's computational effort, you will use a fixed time step for integration. By default, Working Model may decide to compute one animation frame by subdividing it into smaller frames. This feature is useful to ensure accuracy, especially when an object is experiencing a high acceleration. However, in this model, the vehicle is unlikely to undergo a high velocity change since its speed is monitored by a control system.

7. **<u>IMPORTANT:</u> In the Integration Step field of the Simulation Accuracy dialog, click the radio button labeled Fixed.**

Working Model will now always perform one animation frame with a single integration step. This step is necessary to synchronize Working Model with the control system to be implemented in MATLAB. Read on to "Implementing the Control Function" on page 7–16, which provides more discussions on this matter.

Interested readers are encouraged to consult **Appendix A, "Technical Information"** in the *Working Model User's Manual* for an informative discussion on the time step size.

7.3. Creating the Vehicle and Track

The curved track consists of two overlapping curved slots. The vehicle is pinned to the two slots using two offset pins, and the motion of the vehicle is limited to one degree of freedom along the path. In other words, the vehicle can only translate back and forth along the path.

Creating the Track

Working Model constructs a curved slot from a series of control points that you specify. Simply put, you can specify a series of points in the workspace, and Working Model connects them with a smooth curve.

You will not need a numerical geometry of the curve since the precise shape of the curve is not important here. Instead, you can create your own curve with hills and valleys.

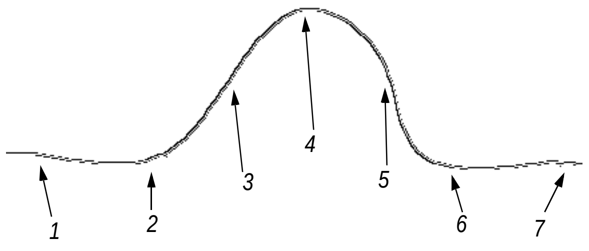
1. **Click the Curved Slot tool in the Toolbar.**

   On MacOS systems, the Curved Slot tool is "hidden" in the Slot pop-up palette by default. Click and hold on the Slot tool in the Toolbar to bring the Slot pop-up palette in view.

2. **Draw a curved slot by clicking a series of control points as shown in Figure 7-2. Six or seven control points should suffice. After clicking the last control point, press the spacebar.**

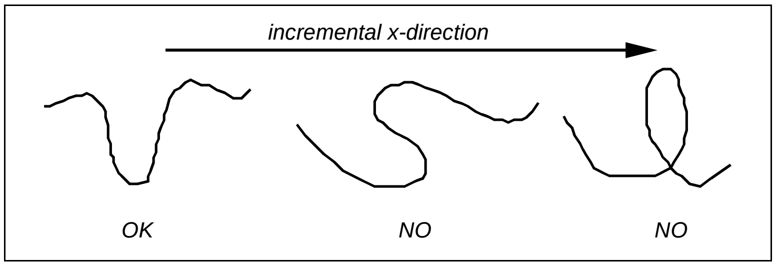
Figure 7-2
Drawing a curved slot

Click in the following order to form a curve (positions are approximate):



NOTE: In this example, you can create a curved slot of your favorite shape, but make sure that the curve is *single valued* with respect to the x-axis. That is, the curve cannot go backward. You will need this restriction because the control system assumes that the vehicle always moves in the incremental x-direction. Figure 7-3 shows what you can and cannot do in the cruise control example.

Figure 7-3
Examples of "Good" and "Bad" curves for the control system



As shown in the previous exercise, you will need to duplicate the curved track since you will attach two pin joints to the vehicle (in order to restrict the rotation).

3. **Select the curved slot you have just drawn and choose Duplicate in the Edit menu.**

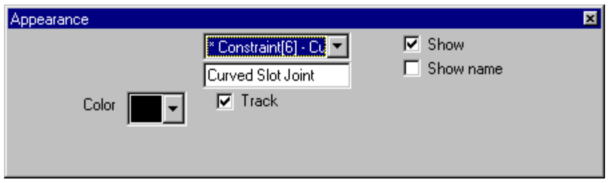
   An identically shaped curved slot appears slightly offset from the original.

Before aligning the two slots, we should name these two elements to identify one from the other.

4. **Select the duplicated slot if it has not been already selected. Choose Appearance in the Window menu.**

   The Appearance window appears. The name field has the default name "Curved Slot Joint" as shown in Figure 7-4.

Figure 7-4
Appearance window for a curved slot



Curved Slot Joint
☑ Track
☑ Show
☐ Show name

5. **Enter Front Point Slot in the name field.**

   We will attach the front point of the vehicle to this slot later.

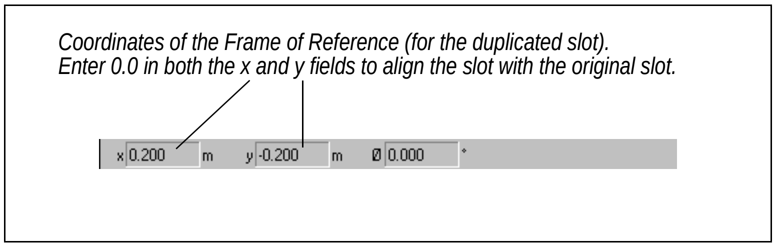
6. **Select the original slot. Enter Center Point Slot in the name field.**

   We will attach the center point of the vehicle to this slot later.

You can align the two slots in one of two ways:

- You can drag the duplicated curved slot so that the curves match. If you have **Grid Snap** activated, you can easily align the two slots.
- Using the Coordinates bar, you can inspect the position of the FOR (frame of reference) of the original slot; it should be (0.0, 0.0). Select the duplicate slot, and type the identical coordinates in the Coordinates bar (as shown in Figure 7-5).

Figure 7-5
FOR of a curved slot as shown in the Coordinates bar


Coordinates of the Frame of Reference (for the duplicated slot).
Enter 0.0 in both the x and y fields to align the slot with the original slot.

x 0.200 m y -0.200 m Ø 0.000 °

7. **Align the two slots using one of the two methods described above.**

Creating the Vehicle

You will use a square object to model the vehicle.

1. **Click the Square tool in the Toolbar.**

   On MacOS systems, the Square tool may be "hidden" in the Rectangle/Square pop-up palette. If the Square tool is not visible in the Toolbar, click and hold on the Rectangle tool to bring the pop-up palette in view.

2. **Draw a square with the sides approximately 1 meter long.**

   The exact size of the square is not important. If you wish, you can use the Coordinates bar to draw precise shapes. While you draw a body, the Coordinates bar continuously displays the dimensions of the object.

You will now create two pins on the vehicle in order to attach the vehicle to the curved tracks.

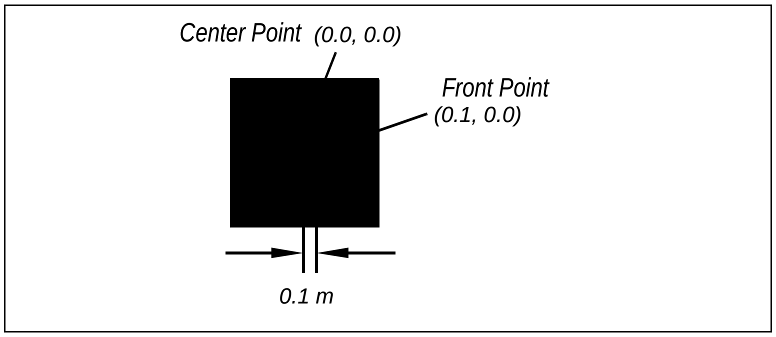
3. **Click the Point tool in the Toolbar.**

4. **Attach a point element to the center of the vehicle.**

   Use the snap point at the center of the square.

5. **Attach another point to the vehicle. Use the Coordinates bar to position the point as shown in Figure 7-6.**

Figure 7-6
Point coordinates on the vehicle



Attaching the Vehicle to the Track

You will now attach the two point elements located on the vehicle to the curved slots.

1. **Using the selection pop-up menu in any utility window, select the Center Point Slot.**

   The selection pop-up menu is located at the top of the Properties, Appearance, and Geometry windows.

2. **Shift-select (select while pressing down the Shift key) the center point on the vehicle.**

   The Join button becomes active.

Join⊙

3. **Click the Join button in the Toolbar.**

   The center point is now attached to the slot element to form a curved slot joint.

4. **Using the selection pop-up menu, select the Front Point Slot.**
5. **Shift-select the front point attached to the vehicle.**

   You may find it easier if you zoom into the picture first by using the Magnifying tool in the toolbar.

The Join button will become active.

Join⊙

6. **Click the Join button in the Toolbar.**

As shown in Figure 7-7, the vehicle should now be attached to the single track. Try dragging the vehicle using the mouse. You may find that dragging this vehicle is not as fast as dragging other objects, since Working Model is constantly computing to maintain the curved slot constraint.

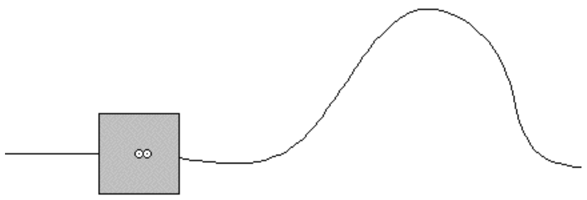


Figure 7-7
Vehicle attached to the slot with two offset pin joints

Oops, My Car is Flipped!

While attaching the second point to the curved slot, you may have accidentally "flipped" the vehicle 180 degrees along the track. That is, the pin attached to (0.1, 0.0) may have actually ended up facing backwards. Should this happen, simply follow the steps shown below. Otherwise, skip ahead to "Implementing the Driving Force" on page 7–11.

1. **If your vehicle has flipped, select the second point (the one located at 0.1 meters in front of the vehicle's center), and click Split in the Toolbar.**

   You are now free to rotate the vehicle.

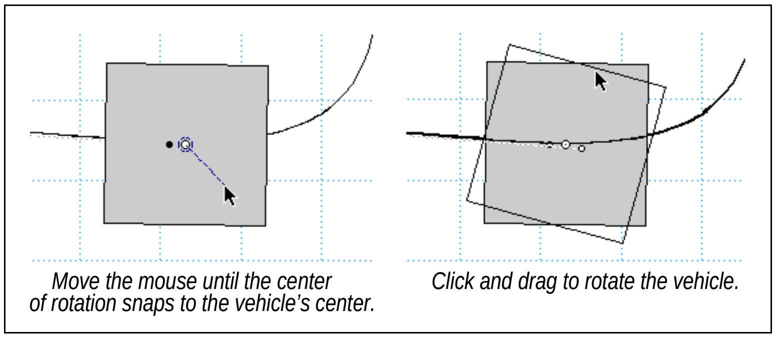
2. **Select the Rotate tool in the Toolbar.**

   As a short cut, you can press the "r" key on your keyboard.

3. **As shown in Figure 7-8, rotate the vehicle around its center until the vehicle is turned about 180 degrees.**

You need not rotate the vehicle exactly 180 degrees. Just bring the second Point so that it is somewhat facing "forward" (to the right) along the track.

Figure 7-8
Rotating the vehicle


Move the mouse until the center of rotation snaps to the vehicle's center.
Click and drag to rotate the vehicle.

4. Click the Join button in the Toolbar.

The second point reattaches to the slot. You did not have to select both the point and the slot to join because Working Model remembered which constraint had been split on the body that was currently selected.

Implementing the Driving Force

To observe how the vehicle moves along the track, you will attach a constant force to drive the vehicle. Expect the vehicle to accelerate at a flat portion of the track, slow down on uphills, and accelerate excessively on downhills.

1. Click the Force tool in the Toolbar.

2. Bring the pointer to the center of the vehicle and find the snap point.

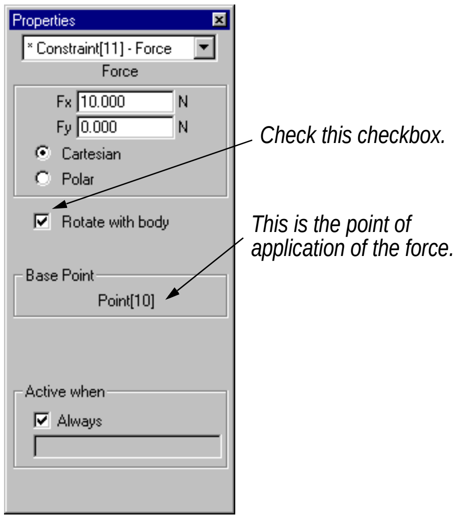
You will see two snap points near the center of the vehicle; one at the geometric center and another at the front point located at (0.1, 0.0). Working Model allows you to snap to any existing point element.

3. Click once when the center snap point is visible and drag the force vector to the left. Click again to complete the force.

4. **In the Coordinates bar, enter 10.0 for the Fx field and 0.0 for the Fy field.**

5. **Open the Properties window and check the box labeled Rotate with body (see Figure 7-9).**

Figure 7-9
Properties window for a force

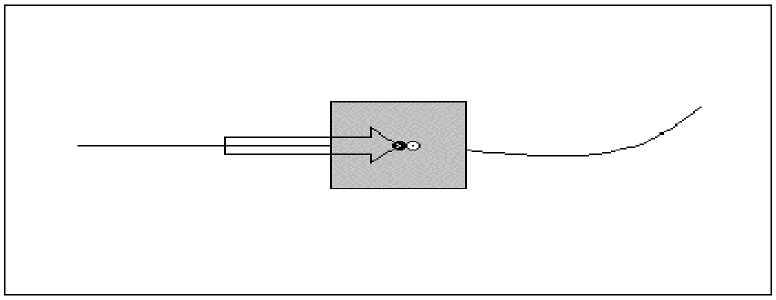


Fx 10.000 N
Fy 0.000 N
[✓ Cartesian] Polar
[✓ Rotate with body]
[✓ Always]

NOTE: The ID numbers of the Force constraint Base Point in your model may be different from the ones shown in Figure 7-9. These numbers are serial counts of all the objects created thus far, and they may differ from one model to another.

The force should be located as shown in Figure 7-10.

Figure 7-10
Vehicle with a force attached



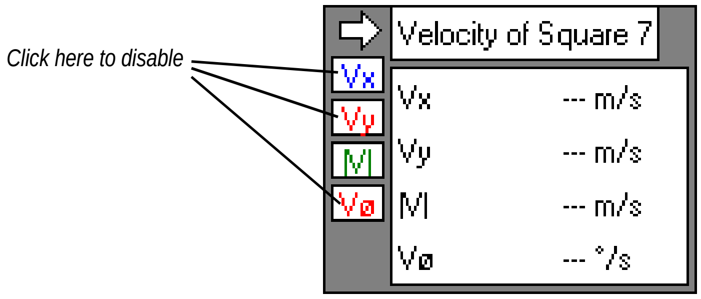
Before starting the simulation, create a meter to measure the speed of the vehicle.

6. Select the vehicle. Choose Velocity in the Measure menu and All from the Velocity submenu.

The meter shows the V_x, V_y, |V| (the speed), and $V_ø$ (angular velocity) of the vehicle.

7. Click the buttons shown in Figure 7-11 to disable unnecessary measurements.

Figure 7-11
Meter for the vehicle's velocity



This meter will serve as an input to the control system. Do not delete it.

8. Click the Run button in the Toolbar.

Observe that the speed of the vehicle does not remain constant as it moves along the hilly road.

For now, let's reset the vehicle to its original position.

9. Click the Reset button in the Toolbar.

7.4. Implementing the Control System

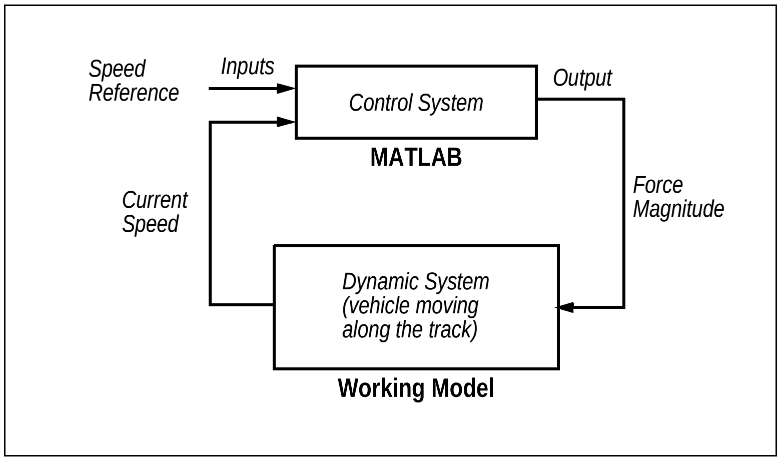
Overview

Any control system can be thought of as a black box with a set of inputs and outputs. The objective of any control system is to maintain a set of controlled properties of a dynamic system. Typically, the inputs to the controller consist of *references* (specified by the user) and *measurements* of the state of the dynamic system. The outputs from the controller are fed to the driving components of the dynamic system.

In this particular example, the controlled property is the speed (*not* the velocity, but its magnitude) of the vehicle. The inputs to the controller are the specified speed of the vehicle (which is the reference) and the speed error (which is a signed difference between the reference and the vehicle's current speed). The output of the control system is the throttle, or the magnitude of the force used to propel the vehicle.

The flow diagram in Figure 7-12 describes the basic concept of our model. Although the implementation will be slightly different, the diagram concisely shows the underlying concept.

Figure 7-12
Flow diagram of the control system



Design Specifics

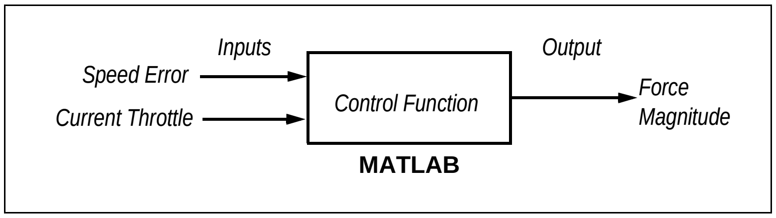
The particular specifications of the control system model are that:

- The user specifies the desired speed in Working Model.
- Working Model measures the current speed of the vehicle, and sends the speed error and the current throttle to a control system implemented in MATLAB.
- Given the current error and throttle, the control system in MATLAB computes the revised throttle and sends it to Working Model.

The control system—which is really a MATLAB function—looks like the following:

Figure 7-13

Schematic diagram of the control function



The control system is implemented with these concepts in mind.

Implementing the Control Function

This example will build on a well-known feedback control scheme called PID control (proportional / integral / derivative). The PID control system is a simple yet relatively robust control system because it takes into account not only the magnitude of an error observed but also its derivative and integral. The PID control system is typically implemented as:

$$u(t) = k_p e(t) + k_i \int e(t) dt + k_d \frac{d}{dt} e(t)$$

where *e(t)* is the error signal, *u(t)* is the actuator signal, and k_p, k_i, k_d are constants that can be tuned for a particular system.

You will implement a discrete version of the PID control system in a MATLAB M-file. If you are a Windows user, the MATLAB M-file called **KRCTRL2.M** is already included in the Tutorial folder that came along in the installation disk.[1]

The following MATLAB script implements the control function shown above in the discrete-time domain.

```
function u = krctrl2(e, u1)
global e1

h=0.01;
kp=10;
ki=.8;
kd=.1;

c1 = kp + (h/2)*ki + (1/h)*kd;
c2 = (h/2)*ki - (1/h)*kd;
c3 = ki;

u = c1*e + c2*e1 + c3*u1;

e1 = e;
```

1. If you chose the Custom Install option, you may have decided to skip this directory/folder. You can install the directory/folder using your original installation CD-ROM.

IMPORTANT: the variable *h* included in the script above is the time step, equivalent to the integration time step used in Working Model. The value of *h* must match the Animation Step as discussed in "Setting Up the Workspace" on page 7–3. If you choose to use a time step of a greater size, you must match the value of *h* in the above script as well. (Recall that you have already set the Animation Step to 0.01 and forced the integration time step to be locked. Do not use variable integration time step in this exercise; otherwise, Working Model may decide to use smaller time steps, in which case the PID control system would become "out of sync" with the computational model.)

Interested users should consult the *MATLAB Reference Guide* for further discussions on the scripting language used in MATLAB. Literature referenced at the end of **Appendix A, "Technical Information"** in the *Working Model User's Manual* further covers implementation of the PID control.

To implement this function:

1. **Launch MATLAB on your computer.**

   Make sure you have MATLAB version 4.2 or greater.

2. **Verify that the directory containing the M-file KRCTRL2.M is in the MATLAB search path. First, locate the directory that contains KRCTRL2.M using Windows Explorer.**

   The M-file KRCTRL2.M is located in the Tutorial directory under the Samples directory in the Working Model installation directory, for example, C:\Program Files\Working Model 2D\Samples\Tutorial. This is the MATLAB search path you will enter in the next step.

3. **In MATLAB, choose Set Path from the File menu.**

4. **Click on [Add Folder...] and choose the directory you have located in step 2.**

5. **Click [Save] and then click [Close].**

   Note that the instructions for step 3 to step 5 have been written for MATLAB Release 12. They might be different for other versions of MATLAB. Consult your MATLAB documentation for more details on setting search paths.

6. **Load the M-file KRCTRL2.M into MATLAB. If you do not have access to the file, create it by typing the above MATLAB script.**

   If you type the MATLAB script manually, make sure to save it in a directory included in the MATLAB search path. See step 2 to step 5 for more details on MATLAB search path.

   You should test to see if the function is properly defined and implemented. Type the following three lines at the MATLAB command window prompt:

   ```
   global e1

   e1 = 1

   krctrl2(1,1)
   ```

   and the function should return the value:

   ```
   ans = 10.8080
   ```

Linking MATLAB with Working Model

Before establishing the link between Working Model and MATLAB, you will create appropriate Control and Meter objects. These objects will serve as the inputs and outputs of the control system.

Creating Inputs and Outputs

The first Control object needed is the one that specifies the desired speed of the vehicle. The value specified in this Control object will be used to compute the current speed error (the difference between the desired speed and the current speed of the vehicle).

1. **Click once on the background to make sure no object is currently selected.**

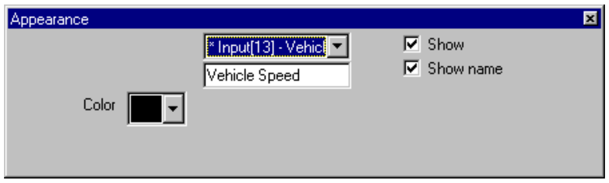
2. **Choose New Control from the Define menu and Generic Control from the submenu.**

   A slider bar appears.

3. **Open the Properties window for the Control object and set the minimum and maximum values to 0.1 and 10.0, respectively.**

To name this Control object:

4. **Open the Appearance window and name the Control object Vehicle Speed. Be sure the Show Name option is checked (see Figure 7-14).**


Appearance

* Input[13] - Vehicl

Vehicle Speed

Color

Show ✓

Show name ✓

Figure 7-14
Appearance window for a control object

You also need to measure the error in vehicle speed.

5. **Choose Time from the Measure menu.**

   A meter showing the time count appears. This meter is a "clock" that shows the elapsed time of the simulation and how each frame corresponds to real time.

The Time meter in itself is not useful for the control system. You will rewrite the definition of the meter so that it measures the speed error.

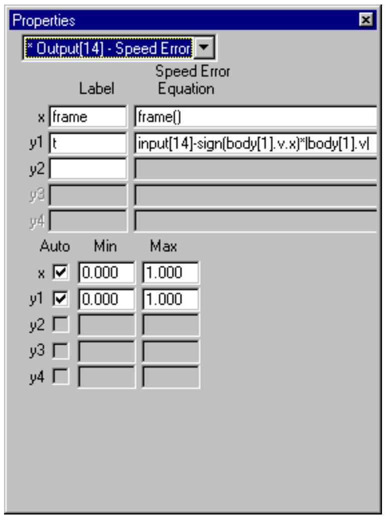
6. **Open the Properties window for the Time meter. Overwrite the Equation field for y1 as follows:**

```
input[##]-sign(body[%%].v.x)*|body[%%].v|
```

where "##" is the object ID number of the speed control you just created and "**%%**" is the object ID number of the square object representing the vehicle. To find out the ID numbers, simply move the pointer over an object, and the Status bar shows you the type of the object and its ID number.

7. **Name the meter Speed Error as you did in step 4.**

The Properties window should resemble Figure 7-15 (enlarge the window for easier viewing).



Properties

* Output[14] - Speed Error

	Label	Speed Error Equation
x	frame	frame()
y1	t	input[14]-sign(body[1].v.x)*|body[1].v|
y2		
y3		
y4		

	Auto	Min	Max
x	✓	0.000	1.000
y1	✓	0.000	1.000
y2			
y3			
y4			

Figure 7-15
Speed Error meter

You will create an input control that will act as a throttle for the force. The value of this input will be calculated in MATLAB and sent to Working Model over the DDE link.

8. Select the force object that is attached to the vehicle. Choose New Control from the Define menu and X-force from the submenu.

Another slider bar appears.

You need a meter measuring the force because our PID control system needs to know the up-to-date magnitude of the force during simulation.

9. Select the Control object for the force and choose Control's Value from the Measure menu.

Both Meter and Control objects are necessary because a Control object in Working Model can only *receive* data from an external application, whereas a Meter object can only *send* data.

Establishing DDE Link with MATLAB

To establish the link with MATLAB:

1. Choose New Application Interface from the Define menu.

A new icon appears in the Working Model workspace.

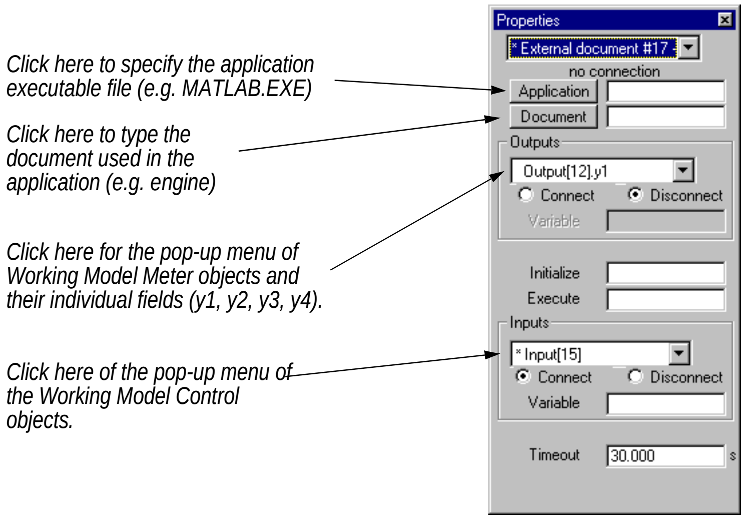
This is an object that specifies a link between the Control/Meter objects of Working Model and variables used in another applications, such as MATLAB.

2. **Select the Interface object you just created and open the Properties window if it is not already open.**

   The Properties window appears (Figure 7-16).

Figure 7-16

Properties window for an external application interface


Click here to specify the application executable file (e.g. MATLAB.EXE)
Click here to type the document used in the application (e.g. engine)
Click here for the pop-up menu of Working Model Meter objects and their individual fields (y1, y2, y3, y4).
Click here of the pop-up menu of the Working Model Control objects.
Properties
* External document #17
no connection
Application
Document
Outputs
Output[12].y1
Connect
Disconnect
Variable
Initialize
Execute
Inputs
* Input[15]
Connect
Disconnect
Variable
Timeout
30.000
s

To let Working Model know that it is going to exchange data with MATLAB:

3. **Click the Application button in the Properties window.**

   A file dialog box appears.

4. **Look for your installation of MATLAB and select MATLAB.EXE in its *bin* directory. Click OK.**

5. **Enter *engine* in the Document field.**

   You want Working Model to exchange data with the MATLAB engine.[1]

To link the Controls and Meters of Working Model with MATLAB variables:

6. **Click on the Output pulldown menu, and select the Meter object that measures the value of the Control object for the force.**

   The name will be something like **output[16].y1**, *although the number 16 may be different in your model. You can find out the ID number using the Status bar.*

7. **Click the Connect radio button and type *u* in the Variable field.**

   The current force measurement is sent to the MATLAB variable **u**.

8. **In the same fashion, establish the links as shown in the following table. Make sure you click the *Connect* radio button for each link.**

Verify the object ID numbers with your model. They may differ in your model.

Description	**Working Model object name**	**MATLAB variable name**
Meter for x-component of the force	output[16].y1	u
Meter for speed error	output[14].y1	err
Control for x-component of the force	input[15]	u

Next, specify the commands to be executed by MATLAB at each simulation step.

9. **In the Initialize and Execute fields of the Properties window for the Application Interface, enter:**

1. For MATLAB, this field should always be *engine;* instead of communicating with a document, Working Model needs to establish a link with the MATLAB engine.

(for Initialize) `u = 0;`

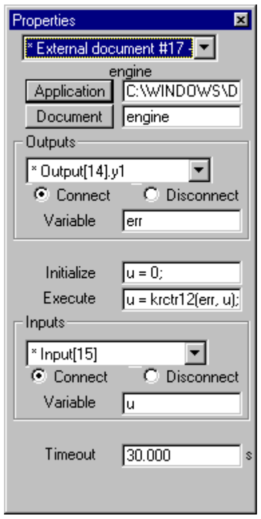
This command tells MATLAB that the force measurement is zero to begin with.

(for Execute) `u = krctrl2(err, u);`

The function call is the implementation of the control system shown in Figure 7-13. MATLAB will execute this function call at every simulation step.

Remember, **krctrl2** is the name of the PID feedback control function mentioned in “Implementing the Control Function” on page 7–16.

Overall, your Properties window for the Application Interface should look like the one in Figure 7-17.



* External document #17

engine

Application: C:\WINDOWS\D

Document: engine

Outputs: * Output[14].y1 — [✓ Connect] Disconnect — Variable: err

Initialize: u = 0;

Execute: u = krctr12(err, u);

Inputs: * Input[15] — [✓ Connect] Disconnect — Variable: u

Timeout: 30.000 s

Figure 7-17
Completed Properties window for the external application interface

You are *almost* ready to run the simulation. Let’s take a break and review what you have done so far. You have:

- set a workspace and accuracy for the simulation
- created a curved track and the vehicle
- attached the vehicle to the track
- implemented the control system

- specified the inter-application link between Working Model and MATLAB

7.5. Running the Simulation

Starting the Simulation

Before you run the simulation, you must initialize a global variable in MATLAB. In your MATLAB Command window, type two commands as follows:

```
global e1;

e1 = 0;
```

These commands are necessary to initialize the state of a variable before you run the simulation.[1]

Now you are ready to proceed.

1. **Adjust the slider bar for the speed control to the desired value.**
2. **Click the Run button in the Toolbar.**

Observe how the vehicle drives along the hill, while the force acts to control the speed of the vehicle. You can change the speed control and to see how fast the PID control system reacts to your command.

Repeating the Simulation

You can click the **Reset** button, and then click the **Run** button again to "playback" what you just saw because *Working Model automatically saves the data from the previous simulation*. You will see the slider bar move up and down on its own, completely mimicking the action you have taken in the previous simulation.

[1]. The variable `e1` is "recycled" in MATLAB as a global variable to store the error term of the previous step in each integration cycle. The scheme is necessary for the discrete PID control function.

If you change anything in MATLAB, you need to erase the simulation history in order to force Working Model to recompute the simulation; Working Model has no way of detecting changes that occur in external applications.

To erase the simulation history:

Reset

1. **Click the Reset button in the Toolbar.**

2. **Choose Start Here from the World menu.**

   Alternatively, you can press Ctrl+H (Windows) or Command+H (MacOS). If the option is dimmed, the simulation history is already erased or you have not run the simulation.

Run ▶

3. **Click the Run button again to start over.**

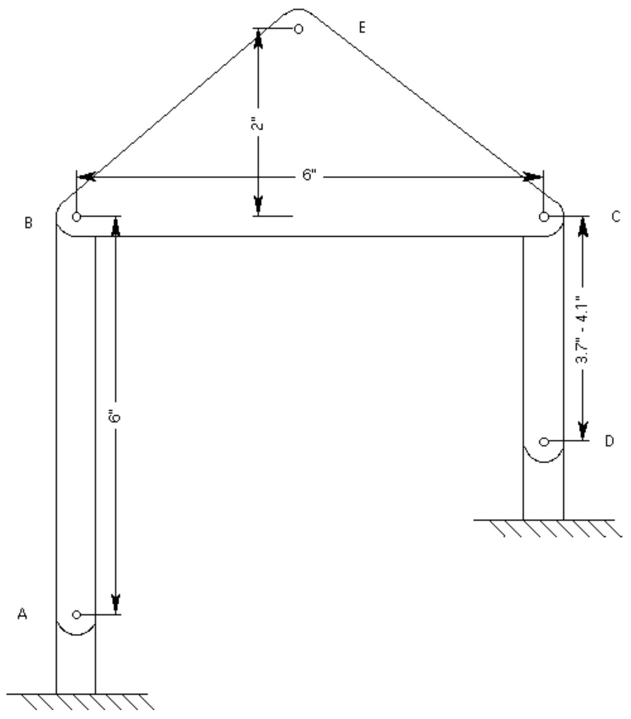
Modifying the Simulation

You may notice that the force initially overcompensates; that is, the speed may deviate slightly from the speed you specified. You can improve the simulation model in several ways as follows:

- Modify the constants k_p, k_i, k_d in the control function to optimize the PID feedback for the particular geometry of the curve or the desired speed.
- Use smaller time steps or use another integration method to run your simulation (see **Appendix A** in the *Working Model User's Manual* for more details on integration methods).
- Implement a completely different control system in MATLAB. The PID control system is only one example of many well-known feedback control laws.

E X E R C I S E 8

Scripting



For the above four-bar linkage, determine the maximum velocities of point E for 5 different lengths of bar CD. Output this data to a file. Bar CD starts at 3.7” and gets incremented by 0.1” each simulation. Bars AB and BC measure 6.0” x 1.0”, and bar CD initially measures 3.7” x 1.0”. All bars are made of steel. The motor at point D applies a constant velocity of 360°/sec. The linkage arms all start at 90° angles to each other. The mechanism moves in a horizontal plane, so you should discount gravity in your calculations.

NOTE: On MacOS systems, the Script Editor requires a PowerPC™ processor. If you are running Working Model on a 680x0-based computer, you will be able to run scripts but will not be able to create or edit them. You can still perform this exercise, however, until the end of **“8.3. Creating the Components”**; then, skip to the note at the end of the next section to run a preinstalled script file which matches the one created in this exercise.

Exercise 8 Concepts:

- Scripting
- Resizing with parametrics
- Output to a file

8.1. Introduction

Often an engineer needs to test a design for a series of different design parameters. An easy way to do this is by running several iterations of analysis with a script in Working Model Basic (WM Basic), Working Model’s embedded scripting language. Scripting can automate the process of running simulations, changing parameters, taking measurements, etc. Anything that can be done manually in Working Model can also be done with a script. Although not addressed in this exercise, you can also modify dialog boxes and create custom interfaces with WM Basic.

This exercise looks at a four-bar linkage, the simplest closed-loop linkage. The four-bar linkage is common in mechanical engineering and is useful for a wide range of engineering applications, from cranes to sprinklers to car hood lifters.

You will learn to write a simple script in WM Basic that runs through a simulation five times, changing one of the components and outputting new data to a file each time. There will be two parts to this exercise. The first shows you how to set up the four-bar linkage interactively. The second shows you how to write a script to manipulate this mechanism.

8.2. Setting Up the Workspace

For this exercise, you will use the English unit system, and the window re-sized to fit the dimensions of our linkage.

1. **Choose Numbers and Units... from the View menu.**

   The Numbers and Units dialog appears.

2. **Choose English (pounds) from the Unit System pop-up menu if it is not already selected (see Figure 8-1).**

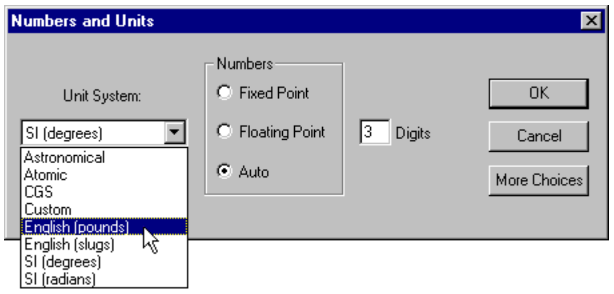


Figure 8-1
Numbers and Units dialog

3. **Click OK.**

4. **Choose View Size... from the View menu.**

   The View Size dialog appears.

5. **Enter 18 in the Window Width field (Figure 8-2).**

   You need to change the window width to accommodate the small dimensions of the linkage.

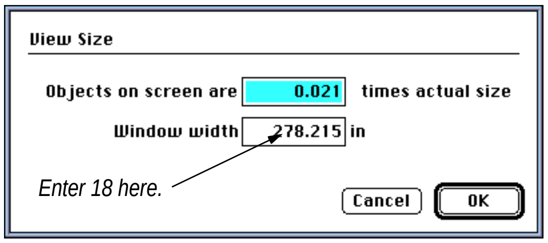


Figure 8-2
View Size dialog

Because the motion of this linkage is constrained to a horizontal plane, we will disregard the effects of gravity.

6. **Choose Gravity... from the World menu.**

   The Gravity dialog appears.

7. **Select None and click OK.**

8.3. Creating the Components

This exercise has three bodies, two 6” x 1” steel bars and a 3.7” x 1” bar. The objects will be created, sized and modified manually in the following steps. Then, a motor will be added to drive the linkage. Finally, we will create an output meter to measure the maximum velocity of point E.

Creating the Bars

The bars will be drawn with the rectangle tool and then resized using the Coordinates bar along the bottom of the screen.

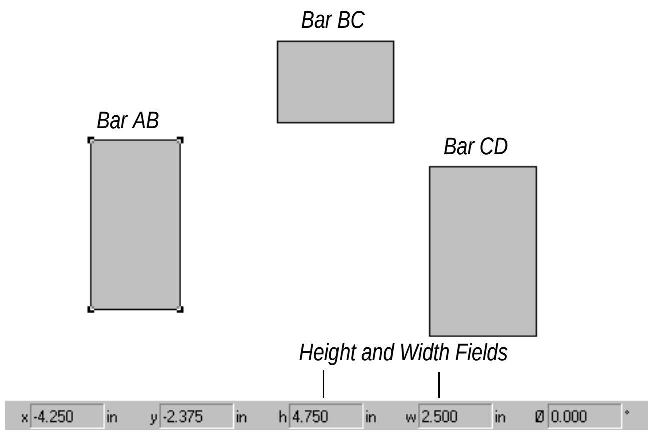
To draw the bars:

1. **Double-click on the Rectangle tool.**

   Double-clicking means that the tool can be used repeatedly without having to re-select it.

2. **Drag out three rectangles: one on the left, one in the center, and one on the right.**

Figure 8-3
Three rectangular bodies: the beginnings of the linkage



x -4.250 in y -2.375 in h 4.750 in w 2.500 in Ø 0.000 °

To size the bars:

1. **Select the Arrow tool.**
2. **Click once on bar AB.**
3. **In the Coordinates bar, set the height of bar AB to 6.0”, and set the width to 1.0” (see Figure 7-3).**
4. **Select bar BC. Set the height to 1.0” and width to 6.0”.**
5. **Select rectangle CD. Set the width to 1.0” and height to 3.7”**

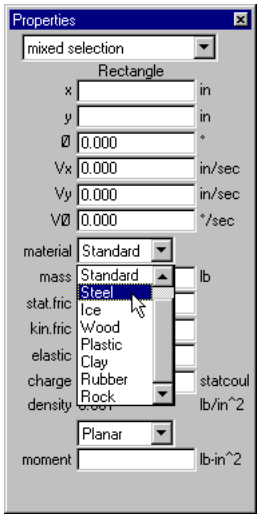
To set the bars’ material to steel:

1. **Double-click on bar AB to open the Properties window.**
2. **Choose Select All from the Edit menu.**

   This highlights all the rectangles so you can edit the properties of all three at once.

3. **Choose steel from the material pop-up menu (see Figure 8-4).**

   Steel is one of the several preset materials in Working Model,



Properties

mixed selection

Rectangle

x: in
y: in
Ø: 0.000 °
Vx: 0.000 in/sec
Vy: 0.000 in/sec
VØ: 0.000 °/sec
material: Standard
mass: lb
stat.fric
kin.fric
elastic
charge: statcoul
density: lb/in^2
Planar
moment: lb-in^2

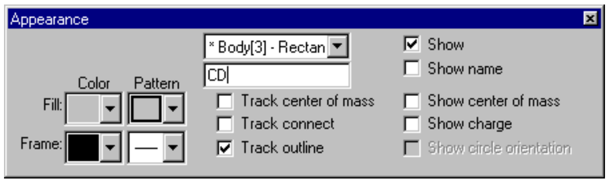
Figure 8-4
Material pop-up menu in the Properties window

Now you will set the name of bar CD. By naming that bar, you can conveniently address it with your script later.

1. **Select the bar CD (the bar on the right side) if not already selected.**

2. **Choose Appearance from the Window menu.**

3. **Enter CD in the name field (Figure 8-5).**



Appearance

* Body[3] - Rectan

CD

Color Pattern
Fill:
Frame:

☐ Track center of mass
☐ Track connect
☑ Track outline

☑ Show
☐ Show name
☐ Show center of mass
☐ Show charge
☐ Show circle orientation

Figure 8-5
Appearance window for a rectangle

Connecting the Bars with Pin Joints

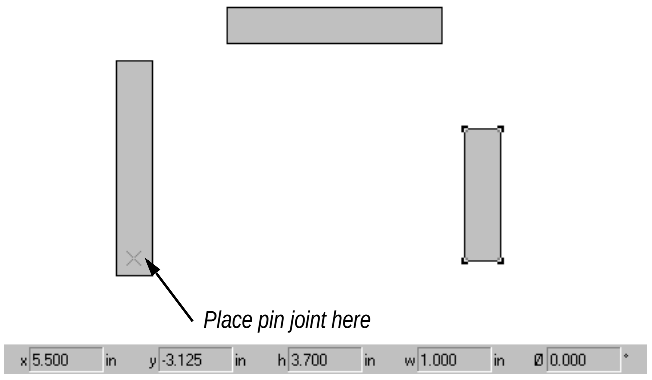
Now you will attach the bars to each other and to the background with pin joints. To pin the left bar to the background:

1. **Click the Pin Joint tool.**

2. **Bring the pointer over point A (Figure 8-6) and find the snap point.**

   An "X" appears near the pointer when you have it properly positioned.

Figure 8-6
Positioning the pin joint



x 5.500 in y -3.125 in h 3.700 in w 1.000 in Ø 0.000 °

3. **When the snap point is visible, click the mouse button.**

   The pin joint attaches the bar to the background.

To connect the bars to each other:

4. **Double-click on the Point tool.**

5. **Using snap points to position each point precisely, create points 1 through 4 as shown in Figure 8-7.**

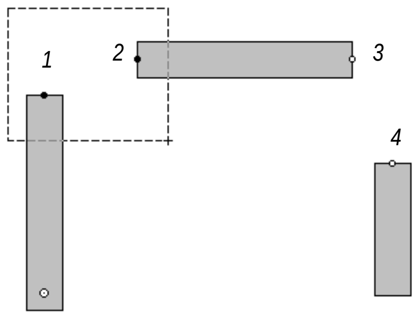


Figure 8-7
Placing the points

6. **Box-select points 1 and 2 as in Figure 8-7.**

   To box-select, first select the Arrow tool. Then, click on the background and drag a box (visible as a dashed line) around the two points to be selected. When the box completely surrounds the points, release the mouse button. The two points should turn black to indicate that they are selected. Also, notice that the Join button becomes active.

Join

7. **Click the Join button in the Toolbar.**

   The two points come together to form a pin joint.

8. **Create another pin joint by performing steps 6 and 7 for points 3 and 4.**

We need to measure the maximum velocity of point E as the length of the short bar changes. To create point E:

9. **Click the Point tool.**
10. **Move the pointer to the center of bar BC (the top bar) and click to create point E when the snap point is visible.**
11. **Double-click point E to open the Properties window.**
12. **In the Y field, enter (2.0) instead of (0.0).**

This offsets point E vertically, 2 inches from bar BC, but keeps it attached to the bar. The dotted line between the point and the rectangle indicates the connection.

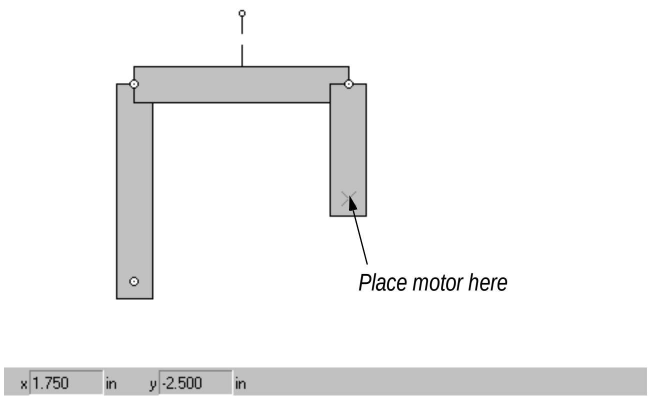
Adding a Motor

To apply a constant torque to our linkage, you need a motor. If the bars were misaligned during joining, you can straighten them by entering 0 in the "ø" box on the Coordinates Bar.

1. **Click the Motor tool.**

2. **Bring the pointer over point D (Figure 8-8) and find the snap point.**

Figure 8-8
Placing the motor



x 1.750 in y -2.500 in

3. **Click when the snap point is visible.**

   The motor connects the bar CD to the background.

The exercise calls for a constant velocity of 360°/sec to be applied to the linkage. To set the motor's velocity:

1. **Double-click on the motor to open the Properties window if it is not already open.**

2. **Enter 360 to set the motor's rotational velocity to 360°/sec.**

Testing Parametrics

Working Model's parametrics feature will automatically rebuild a model for you after any change in the design. To see how parametrics works, we will resize one of the bars and watch Working Model automatically update the design while maintaining all connection and dimension constraints.

1. **Select bar AB (the left bar).**

2. **Change its height to 5.0" in the Coordinates bar and press Return or Enter.**

   Working Model automatically rebuilds your model. Notice that the relative positions of constraints such as pin joints and motor are preserved.

3. **Change the height of bar AB back to 6.0".**

Adding an Output Meter

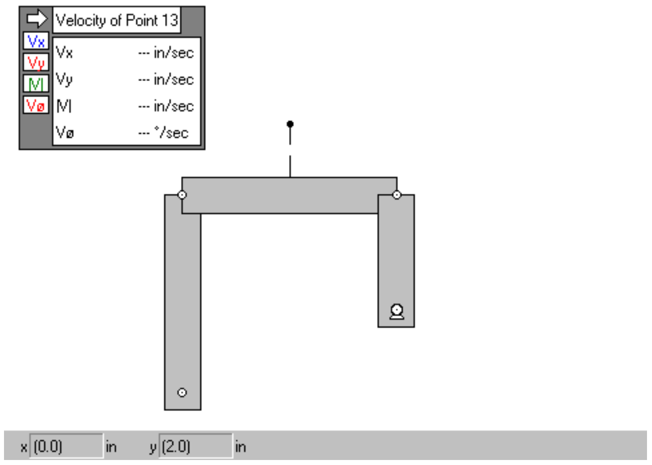
To measure the maximum velocity of point E, we need an output meter. The output meter will show both the current velocity of the point and the maximum velocity of the point throughout the run. Later we will use the script to take the maximum velocity from the meter and output it to a file.

1. **Click point E once to select it.**

2. **Choose Velocity from the Measure menu, and All from the Velocity submenu.**

   A meter appears on the screen (see Figure 8-9).

Figure 8-9
Creating the meter



3. **Double-click on the meter.**

   The Properties window for the meter appears. We will now overwrite some of the fields in the meter.

4. **Delete the label Vx in the row marked y1 and press Return or Enter.**

   The equation field disappears along with the label field, and the rest of the meter columns shift upward.

5. **Delete the label Vy and press Return or Enter.**

6. **Change the label field that reads Vø to Vmax.**

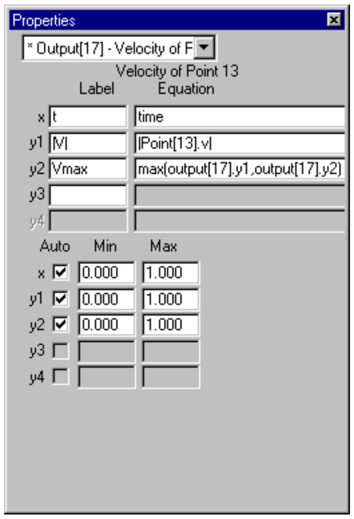
7. **Note the output number (output[N]) at the top of the Properties window.**

   Use this output number N in the next step.

8. **Delete the equation field for Vmax.**

9. **Now type: max(output[N].y1, output[N].y2) and press return.**

Where you see a "N" above, type the output number you noted in the previous step. The max statement you just typed (shown in Figure 8-10) tells the output meter to display the maximum value of y1, the current velocity, and y2, the previous maximum velocity. In other words, this statement keeps track of the point's maximum velocity.



	Label	Equation
x	t	time
y1	|V|	|Point[13].v|
y2	Vmax	max(output[17].y1,output[17].y2)
y3		
y4		

	Auto	Min	Max
x	✓	0.000	1.000
y1	✓	0.000	1.000
y2	✓	0.000	1.000
y3			
y4			

Figure 8-10
Formula for tracking the max velocity

10. Close the Properties window.

11. If not already open, choose Appearance from the Windows menu.

12. Enter Vmax in the name field.

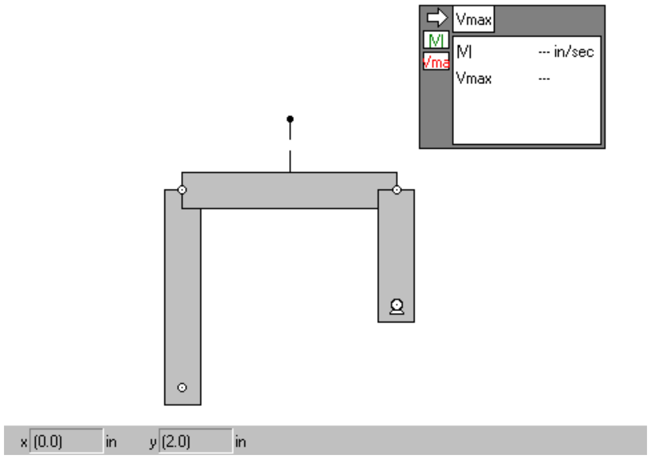
This names the meter "Vmax" so we can access it conveniently from the script we write later.

13. Close the Appearance window.

14. Drag the meter to a desired location.

Your window should now resemble Figure 8-11.

Figure 8-11
Completed linkage



Running the Simulation

To run the simulation:

1. **Click the Run button in the Toolbar.**

   Note when the maximum velocity is attained.

2. **Click Stop once a complete loop is run.**

   On MacOS systems, the Run button turns into the Stop button while the simulation is running.

3. **Click Reset.**

If you are interested, use the tape player controls at the bottom of the document window to advance simulation frames. Try to find out at which frame the maximum velocity was attained.

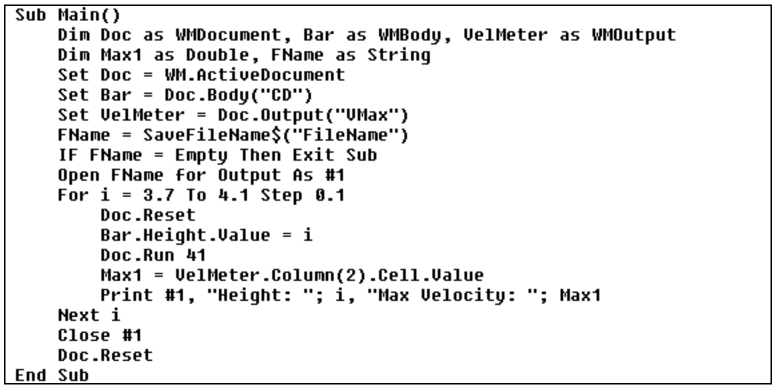
8.4. Automating the Process

Writing a WM Basic Script

WM Basic allows you to automate any process that you would normally handle manually in Working Model (you can also modify dialog boxes and create custom interfaces). In this case, we will use WM Basic to run the simulation 5 times while varying a single parameter and gathering data in each run. Automating this process with WM Basic is much more convenient and time-effective than repeatedly altering the simulation manually.

The first thing you need to do is open a new script to write your code in. All scripting commands need to be entered on separate lines, so each time you're told to enter some text, do so on a new line. When finished, your script should match Figure 8-12.

Figure 8-12
WM Basic script



```
Sub Main()
    Dim Doc as WMDocument, Bar as WMBody, VelMeter as WMOutput
    Dim Max1 as Double, FName as String
    Set Doc = WM.ActiveDocument
    Set Bar = Doc.Body("CD")
    Set VelMeter = Doc.Output("VMax")
    FName = SaveFileName$("FileName")
    IF FName = Empty Then Exit Sub
    Open FName for Output As #1
    For i = 3.7 To 4.1 Step 0.1
        Doc.Reset
        Bar.Height.Value = i
        Doc.Run 41
        Max1 = VelMeter.Column(2).Cell.Value
        Print #1, "Height: "; i, "Max Velocity: "; Max1
    Next i
    Close #1
    Doc.Reset
End Sub
```

1. **Select Editor from the Script menu.**

   This automatically creates a new script for you to edit.

Before the script can access any of the Working Model objects in your simulation, you must specify which objects you want to address. You do this by creating variables that hold the names of the objects. In this case, we're concerned with: a Working Model document, one rectangle body, one output meter (that contains the max velocity), and the file to which data is output.

2. **Beneath "Sub Main()" indent one tab and enter: Dim Doc as WMDocument, Bar as WMBody, VelMeter as WMOutput**

   The statement defines a variable called Doc of type WMDocument. Similarly, Bar and VelMeter are defined as variables of types WMBody and WMOutput, respectively.

   Later, we will assign the bar CD to the variable Bar. This way, we will be able to modify the length of the bar CD by modifying the corresponding property of the variable Bar. Also, we will assign the existing meter object to the variable VelMeter.

3. **Enter: Dim Max1 as Double, FName as String**

   The statement defines one variable, Max1, of type Double, and a second, FName, of type String. A double is a double-precision floating point number.

   The variable FName will hold the name of the file to which you will output data.

   The types Double and String are standard BASIC types, whereas WMDocument, WMBody, and WMOutput are types specifically defined for WM Basic.

4. **Enter: Set Doc = WM.ActiveDocument**

   A few lines above, we created a variable called Doc. Now we assign it a value. This assignment tells WM Basic that Doc refers to the currently active document.

5. **Enter: Set Bar = Doc.Body("CD")**

   This assignment tells WM Basic that Bar refers to the bar we labeled CD.

6. **Enter: Set VelMeter = Doc.Output("Vmax")**

   This tells WM Basic that VelMeter refers to the output meter we named Vmax.

7. **Enter: FName = SaveFileName$("Filename")**

This statement calls up a dialog box that prompts you for a filename to which the Working Model data should be saved. The string, `FName`*, stores the filename you select.*

8. **Enter: If FName = Empty Then Exit Sub**

   If the file you selected is empty (which only happens when you click "Cancel" in the dialog box), then the script terminates.

9. **Enter: Open FName for Output As #1**

   This statement opens the file you selected in step 11. We will write our data to this file.

Now you will write a loop that tells Working Model to process the simulation for bar lengths of 3.7" to 4.1" increments of 0.1".

10. **Enter: For i = 3.7 To 4.1 Step 0.1**

    This creates a loop that starts counting at 3.7 and goes up by 0.1 until it reaches 4.1. The variable i is the "counter" that keeps track of the loop's progress.

11. **Indent one tab and enter: Doc.Reset**

    This resets the simulation prior to each run.

12. **Enter: Bar.Height.Value = i**

    This sets the height of the bar to i, which will initially equal 3.7 and go up by 0.1 until it reaches 4.1.

13. **Enter: Doc.Run 41**

    The statement runs the simulation for 41 frames, enough frames so that bar CD completes one full revolution.

14. **Enter: Max1 = VelMeter.Column(2).Cell.Value**

    This assigns the value of the velocity meter's second field, the Vmax field, to Max1. The velocity meter's Vmax field contains the max velocity of the point defined by the equation we entered earlier.

15. **Enter: Print #1, "Height: "; i, "Max Velocity: "; Max1**

This statement outputs the maximum velocity to the file created in the SaveFileName statement earlier. We refer to it here with the ID, #1 (assigned with the Open statement earlier).

The variable i is the height of the bar, and Max1 is the maximum velocity. The semi-colon tells WM Basic to print the value immediately after the previous value; a comma says to print in the next field (each field is 14 spaces).

16. **Enter: Next i**

    This increments the variable i by one step, in this case 0.1, and sends the program back to the start of the for loop.

17. **Enter: Close #1**

    This closes the data file. Again we refer to the file by the ID, #1, that we gave it when we opened it.

18. **Enter: Doc.Reset**

    This resets the simulation.

The script is finished, but you should save it before you run it.

19. **Choose Save from the File menu in the Script Editor window.**

    Name the file and click OK.

Running the Script

You are now ready to run the script.

1. **Choose Start from the Run menu in the Script Editor window or click the Start button in the Script Editor Toolbar.**

NOTE: The script created in this exercise is included as a file called **Scripting.WBS** in the Tutorial folder which was installed with Working Model. Users of 680x0-based MacOS systems (who cannot use the Script Editor) can still follow this exercise by running the script file.

To run the preinstalled script file:

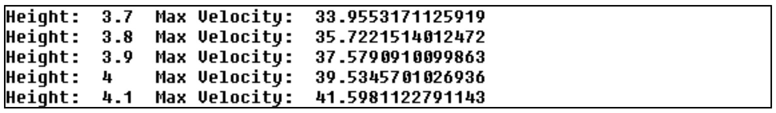
1. **Choose Run... from the Script menu.**

   A file browsing dialog appears.

2. **Open the Tutorials folder and select the file Scripting.WBS.**
3. **Click Open.**

After you run your script, take a look at the output data file using a simple text editor such as Notepad (on Windows systems) or SimpleText (on MacOS systems). It should resemble Figure 8-13.

Figure 8-13
Scripting.txt file



```
Height:  3.7  Max Velocity:  33.9553171125919
Height:  3.8  Max Velocity:  35.7221514012472
Height:  3.9  Max Velocity:  37.5790910099863
Height:  4    Max Velocity:  39.5345701026936
Height:  4.1  Max Velocity:  41.5981122791143
```

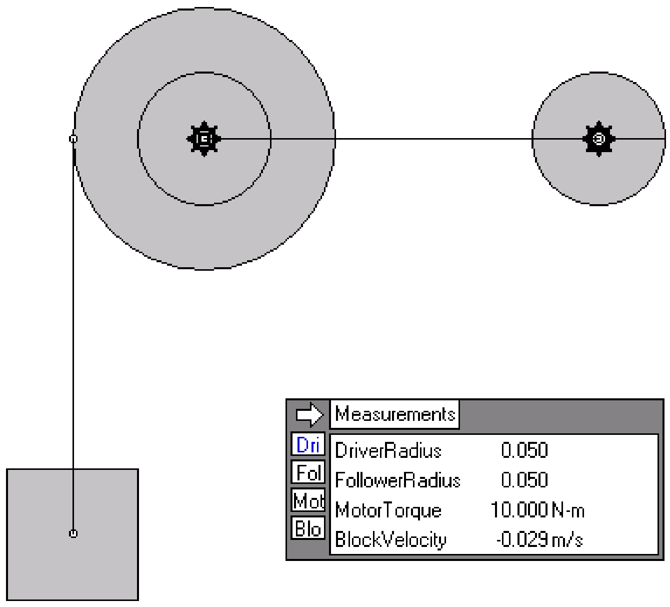
Modifying the Script

Go ahead and experiment with your script. Try increasing the range of the bar’s height. Try modifying two bars, instead of one. Output other data to the file.

This exercise should give you a preliminary idea of how WM Basic works. For more detailed information, consult the*Working Model Basic User’s Manual*.

E X E R C I S E   9

Advanced Scripting



DriverRadius	0.050
FollowerRadius	0.050
MotorTorque	10.000 N-m
BlockVelocity	-0.029 m/s

You have been given a selection of motors and gears and asked to design a mechanical system that will provide enough torque to lift the block. Determine what size of motor you need (the amount of torque it can output) and the gear ratio of the system. The wheel and the gears and made of steel. The wheel has a radius of 0.1 m. There are five sizes of gears from which to choose: 0.01 m, 0.02 m, 0.03 m, 0.04 m, and 0.05 m. The mass of the block is 10 kg.

NOTE: On MacOS systems, the Script Editor requires a PowerPC™ processor. If you are running Working Model on a 680x0-based computer, you will be able to run scripts but will not be able to create or edit them. You can still perform this exercise, however, until the end of **“9.3. Creating the Components”**; then, skip to the note at the end of the next section to run a preinstalled script file which matches the one created in this exercise.

Exercise 9 Concepts:

- Parametrizing a constraint
- Pause control
- WM Dialog box
- Script button

9.1. Introduction

In the exercise, you will create a motor and gear system that lifts the block. By running through the simulations with different motor torques and gear sizes, you will see what design options are available.

You will write a script in WM Basic with your own dialog box. This script will allow you to input the sizes of motors you have. It then runs through the simulation multiple times, changing the driver gear, follower gear, and the motor torque. Finally, it outputs data to a file. There are two parts to this exercise. The first shows you how to set up the mechanical system. The second shows you how to write a script to run this mechanism.

Note: The focus of this exercise is advanced scripting. If you are not comfortable with drawing objects, adding constraints, or using other simple Working Model features, please review the previous tutorials. Also, if you have not done the previous scripting tutorial, now is a good time to do it.

9.2. Setting Up the Workspace

Before starting the exercise, change a few settings.

1. **Choose Numbers and Units... from the View menu.**

2. **Choose SI (degrees) from the Unit System pop-up menu and click [OK].**

3. **Choose View Size... from the View menu and enter 1.2 m in the Window Width field.**

   You need to change the window width to accommodate the small dimensions of the system.

4. **Choose Accuracy... from the World menu.**

5. **Click the radio button by the frame rate numbers and enter 0.005 in the sec (top) field or enter 200 in the /sec (bottom) field. Click [OK].**

6. **Choose Pause Control... from the World menu.**

   The Pause Control dialog appears,Figure 9-1.

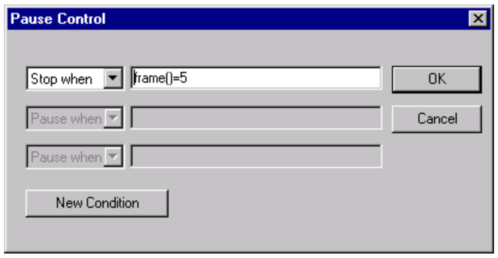


Figure 9-1
Pause Control dialog

7. **Click the New Condition button**

8. **Select Stop when from the pull-down menu, enter frame()=5 in the field next to it, and click [OK].**

   Pause control allows you to specify the duration of the simulation. In this case, we are only interested in the first five frames of the simulation.

9.3. Creating the Components

This exercise has four bodies: a wheel of radius 0.1 m, a driver gear of radius 0.05 m, a follower gear of radius 0.05 radius, and a 0.1 m x 0.1 m block of mass 10 kg. These bodies will be created, sized, and modified manually in the following steps. Then, a motor will be added to drive the driver gear. A gear will connect the driver gear to the follower gear.

Creating the Wheel

1. **Draw a circle.**
2. **Name the circle "Wheel".**
3. **Set its material to steel.**
4. **Set its radius to 0.1 m and locate it at (0.0, 0.0).**

Creating the Driver Gear

1. **Draw a circle and name it "Driver".**
2. **Set its material to steel.**
3. **Set its radius to 0.1 m and locate it at (0.3, 0.0).**

Creating the Follower Gear

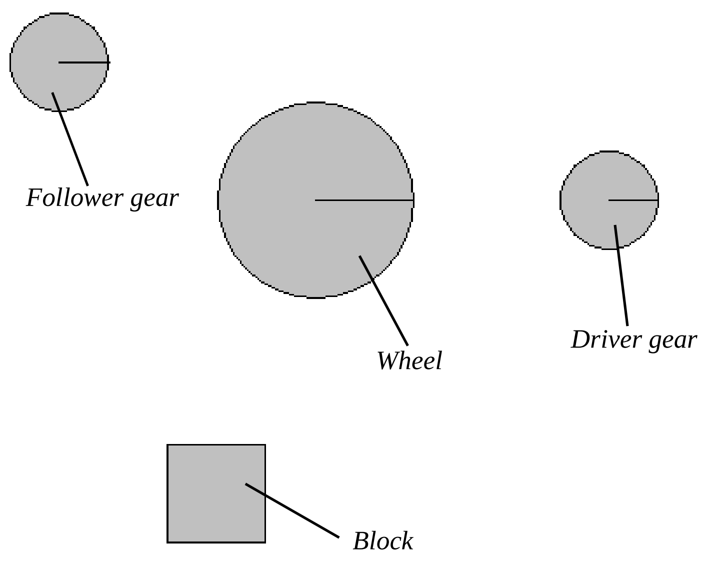
1. **Draw a circle and name it "Follower".**
2. **Name the circle Follower.**
3. **Set its material to steel.**

The Follower is resized and relocated later.

Creating the Block

1. **Draw a square and name it "Block".**
2. **Set its height and width to 0.1 m and locate it at (-0.1, -0.3).**
3. **Change its mass to 10 kg.**

Figure 9-2
The components



9.4. Adding Constraints to the Components

Attaching the Wheel to the Background with a Pin Joint

To pin the Wheel to the background with a pin joint:

1. **Click the Pin Joint tool.**
2. **Attach the pin joint to the center of the wheel.**

Attaching the Follower to the Wheel with a Rigid Joint

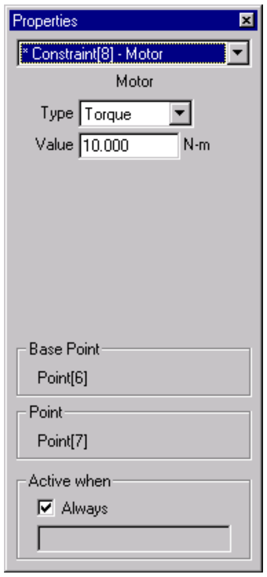
To size and relocate the Follower:

1. **Select the Follower.**
2. **Set its radius 0.05 m and locate it at (0.0, 0.0).**
3. **Click the Rigid Joint tool.**
4. **Attach to the rigid joint to the center of the Follower.**

Adding a Motor to the Driver

To apply a constant torque to the gear system, add a 10 N-m torque motor to the Driver.

1. **Click the Motor tool.**
2. **Attach the motor to the center of the Driver.**
3. **Double-click on the motor to open the Properties window (if it is not already open). See Figure 9-3.**
4. **Under the Type pull-down menu, select Torque and enter the value 10 set the motor's torque to 10 N-m.**


Properties

* Constraint[8] - Motor

Motor

Type Torque

Value 10.000 N-m

Base Point

Point[6]

Point

Point[7]

Active when

☑ Always

Figure 9-3
Properties window for the driver motor

Connecting the Follower to the Driver

To transmit the torque from the Driver’s motor to the Follower and the Wheel, connect the Follower to the Driver with the gear tool as follows:

1. **Choose the Gear tool.**
2. **Click anywhere on the Follower.**
3. **Move the pointer to the Driver and click the mouse button again.**

Connecting the Block to the Follower with a Rope

We connect the Block to the Follower with the rope tool. In the real world, the rope wraps around the Wheel as the Block is lifted. In the model, we simulate that behavior by parametrizing the length of the rope.

To add the rope:

1. **Choose the Rope tool.**
2. **Click on othe exact center of the block.**

Note: Working Model assists your choosing the exact center of the block if you hover the mouse-pointer over the center of the block until the snap point (x symbol) appears.

3. **Move the loose end of the rope over the center of the Follower and find the snap point.**
4. **Click when the snap point is visible.**

To parametrize the length of the rope:

1. **Double-click on the Wheel. Note the body number (Body[M]) at the top of the Properties window.**

   Use this body number M in the steps below.

2. **Double-click on the Block. Note the body number (Body[N] at the top of the Properties window.**

   Use this body number N in the steps below.

3. **Double-click on the rope.**
4. **In the Length field in the Properties window, type: abs(body[M].p.y-body[N].p.y) and press Enter.**

   The workspace should resemble Figure 9-4.

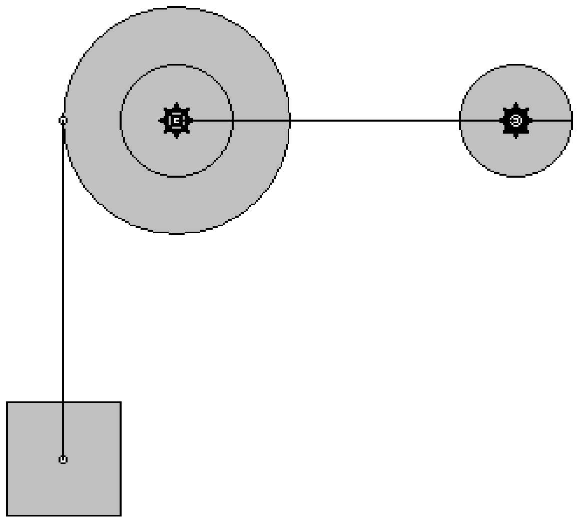


Figure 9-4
Adding constraints to the components

9.5. Adding an Output Meter

To determine whether a particular combination of motor and gears produces enough torque to lift the block, we will need to measure the driver radius, follower radius, motor torque, and block velocity.

First, we need to know the ID numbers that are used to designate the bodies and constraints. To determine these ID numbers:

1. **Double-click on the Driver. Note the body number (Body[A]) at the top of the Properties window.**

   Use this body number A in the steps below. If the body is too small to be easily selected, double-click on any object to open the Properties window. Then select the body you want, in this case Driver, from the pull-down menu at the top of the Properties window.

2. **Double-click on the Follower. Note the body number (Body[B]) at the top of the Properties window.**

Use this body number B in the steps below.

3. **Double-click on the Motor. Note the constraint number (Constraint[C]) at the top of the Properties window.**

   Use this constraint number C in the steps below.

4. **Double-click on the Block. Note the body number (Body[D]) at the top of the Properties window.**

   Use this body number D in the steps below.

To create the output meter:

1. **Click on the Block.**

2. **Choose Velocity from the Measure menu, and Y graph from the Velocity submenu.**

   A meter appears on the screen.

3. **Double-click on the meter.**

   The Properties window for the meter appears. We will now overwrite some of the fields in the meter.

4. **Delete the label Vy in the row marked y1 and press Enter.**

   The equation field disappears along with the label field, and the rest of the meter columns shift upward.

To enter labels and equations:

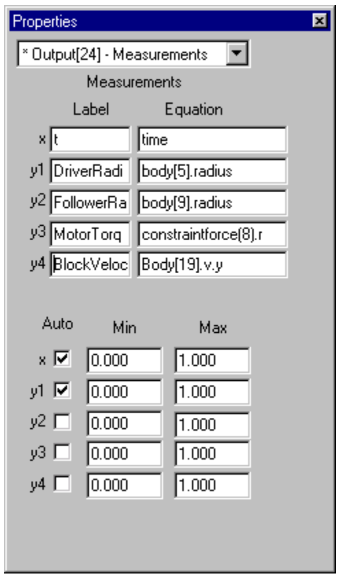
5. **Click on the label field on the row marked y1, enter DriverRadius, and press Enter. Then, click on the associated equation field and enter body[A].radius.**

6. **Click on the label field on the row marked y2, enter FollowerRadius, and press Enter. Then, click on the associated equation field and enter body[B].radius.**

7. **Click on the label field on the row marked y3, enter MotorTorque, and press Enter. Then, click on the associated equation field and enter constraintforce[C].r.**

8. **Click on the label field on the row marked y4, enter BlockVelocity, and press Enter Then click on the associated equation field and enter body[D].v.y.**

   The Properties window should resemble Figure 9-5.



Properties

* Output[24] - Measurements

Measurements

	Label	Equation
x	t	time
y1	DriverRadi	body[5].radius
y2	FollowerRa	body[9].radius
y3	MotorTorq	constraintforce[8].r
y4	BlockVeloc	Body[19].v.y

	Auto	Min	Max
x	✓	0.000	1.000
y1	✓	0.000	1.000
y2	☐	0.000	1.000
y3	☐	0.000	1.000
y4	☐	0.000	1.000

Figure 9-5
Properties window for the output

9. **Close the Properties window.**

To change the name of the output meter:

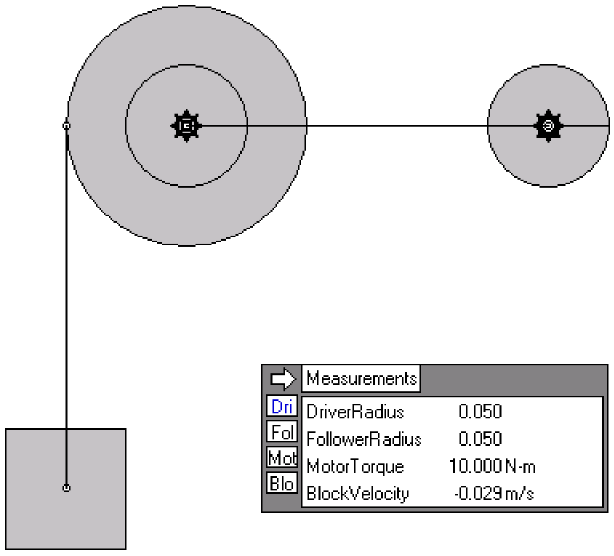
1. **Click on the output meter you just created.**
2. **Choose Appearance from the Windows menu and enter “Measurements” in the name field.**

   The meter name “Measurements” will be used in a script shortly.

3. **Close the Appearance window.**

Your window should now resemble Figure 9-6.

Figure 9-6
Completed gear system



```
DriverRadius      0.050
FollowerRadius    0.050
MotorTorque      10.000 N-m
BlockVelocity    -0.029 m/s
```

Running the Simulation

To run the simulation:

1. **Click the Run button in the Toolbar.**
2. **Click Reset.**

9.6. Automating the Process

Writing a WM Basic Script

We will use WM Basic to run the simulation multiple times while varying three parameters and gathering data in each run. These three parameters are driver gear radius, follower gear radius, and motor torque. Automating this process with WM Basic is more convenient and time-effective than repeatedly altering the simulation manually.

1. **Select Editor from the Script menu.**

This automatically creates a new script for you to edit.

2. **Before we write the script, we will first create a useful component of WM Basic, the dialog box, which we will incorporate into our script.**

Creating a Dialog Box

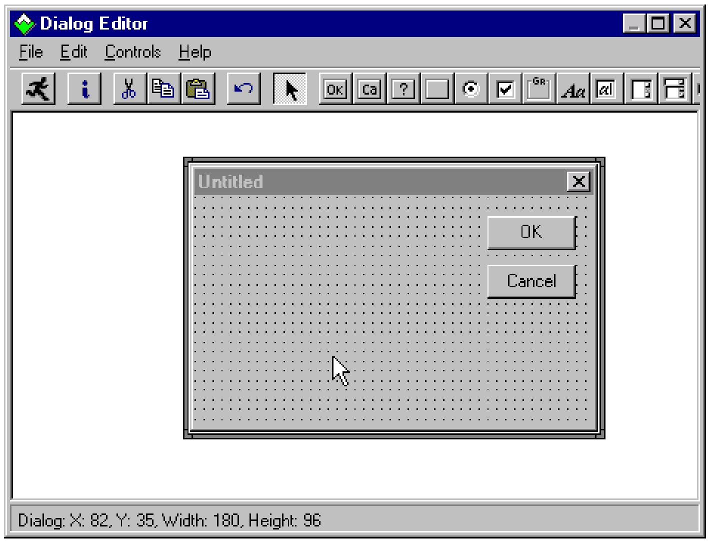
You can create custom dialog boxes, which can obtain information from the users. In this case, you want to ask the users what range of torques their motors can generate. We will then use this information to run the script based on those values.

We will create a dialog box that will ask the users to input the following information: the maximum motor torque, the minimum motor torque, and the iteration step.

To add a Dialog Box:

1. **In the Script Editor, select Insert New Dialog... from the Edit menu.**

   This opens the Dialog Editor. A new dialog box titled "Untitled" appears in the editor. See Figure 9-7.



Dialog: X: 82, Y: 35, Width: 180, Height: 96

Figure 9-7
Dialog Editor

2. **Select Text from the Controls menu.**
3. **Click on anywhere on the dialog box.**

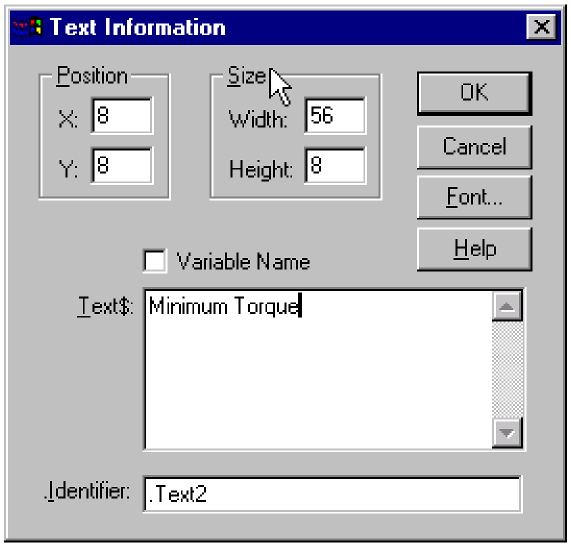
   The grids in the dialog box help you locate the text object.

4. **Double-click on the text.**

   A Text Information window appears.

5. **In the Text$ field, enter Minimum Torque.**
6. **In the Position frame, enter 8 in the X field and 8 in the Y field.**
7. **In the Size frame, enter 56 in the Width field and Height field and click [OK].**

   The Text Information window should resemble Figure 9-8.



X: 8
Y: 8
Width: 56
Height: 8
Text$: Minimum Torque
Identifier: .Text2

Figure 9-8
Text Information window for Minimum Torque

8. **Select Text box from the Controls menu.**
9. **Click on an area just below the text Minimum Torque.**
10. **Double-click on the text box.**

A Text Box Information window appears.

11. **In the .Identifier field, enter .MinimumTorque.**

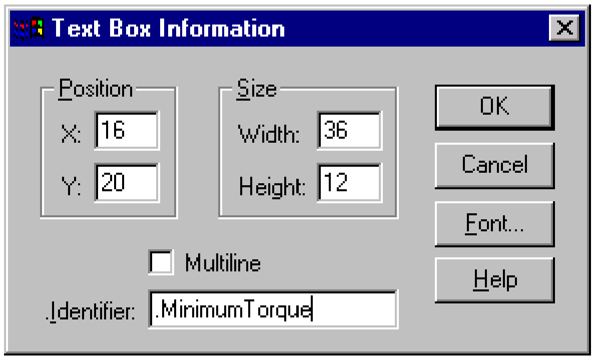
    We will use the name .MinimumTorque later in the script.

12. **In the Position frame, enter 16 in the X field and 20 in the Y field.**

13. **In the Size frame, enter 36 in the Width field and 12 in the Height field and click [OK]..**

    The Text Box Information window should resemble Figure 9-9.

Figure 9-9
Text Box Information window for Minimum Torque



Position
X: 16
Y: 20

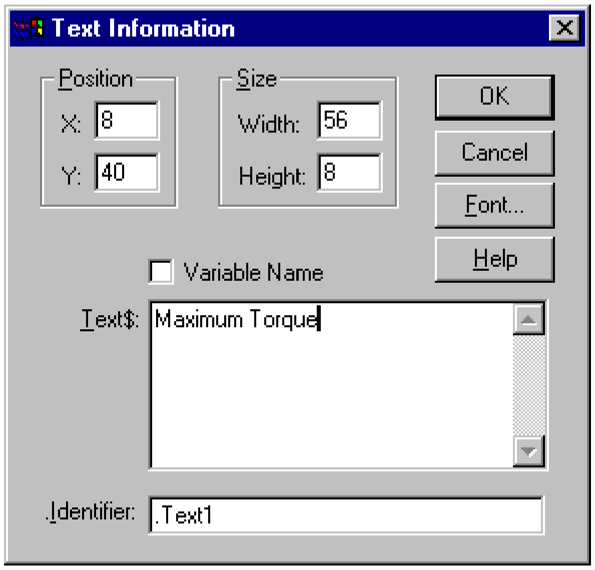
Size
Width: 36
Height: 12

☐ Multiline

.Identifier: .MinimumTorque

14. **Follow step 3 to step 8 to create a text object for the label Maximum Torque.**

Figure 9-10
Text Information window for Maximum Torque



Position
X: 8
Y: 40

Size
Width: 56
Height: 8

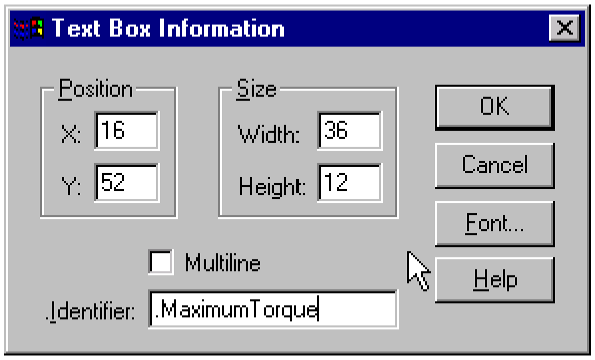
Variable Name [ ]

Text$: Maximum Torque

Identifier: .Text1

15. Follow step 9 to step 14 to create a text box object for the input Maximum Torque.

Figure 9-11
Text Box Information window for Maximum Torque



Position
X: 16
Y: 52

Size
Width: 36
Height: 12

Multiline [ ]

Identifier: .MaximumTorque

16. Follow step 3 to step 8 to create a text object for the label Iteration Step.

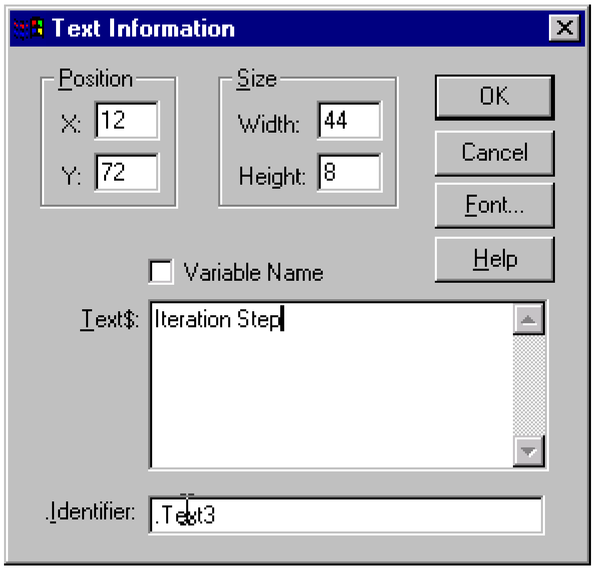


Figure 9-12
Text Information window for Iteration Step

Text Information

Position
X: 12
Y: 72

Size
Width: 44
Height: 8

☐ Variable Name

Text$: Iteration Step

Identifier: .Text3

17. Follow step 9 to step 14 to create a text box object for the input Iteration Step.

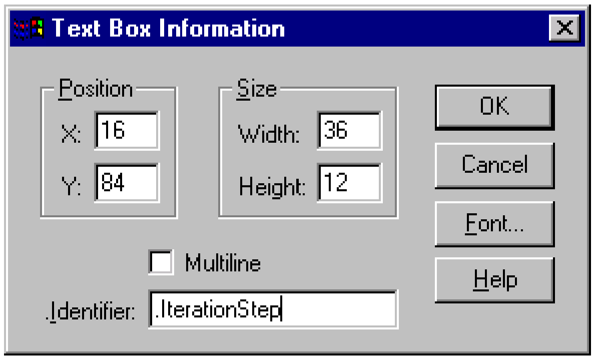


Figure 9-13
Text Box Information window for Iteration Step

Text Box Information

Position
X: 16
Y: 84

Size
Width: 36
Height: 12

☐ Multiline

Identifier: .IterationStep

18. Double-click on the title bar of the dialog box.

The Dialog Box Information window appears, Figure 9-14.

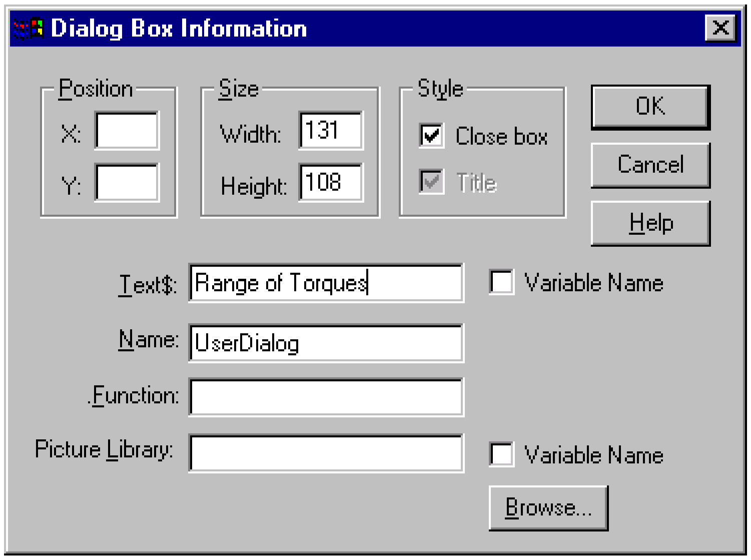


Figure 9-14
Dialog Box Information window

Dialog Box Information

Position
X:
Y:

Size
Width: 131
Height: 108

Style
[✓] Close box
[✓] Title

OK
Cancel
Help

Text$: Range of Torques
[ ] Variable Name

Name: UserDialog

.Function:

Picture Library:
[ ] Variable Name

Browse...

19. Delete Untitled from the Text$ field, enter Range of Torques, and click [OK].

This assigns the name "Range of Torques" to the dialog box.

The dialog box should resemble Figure 9-15.

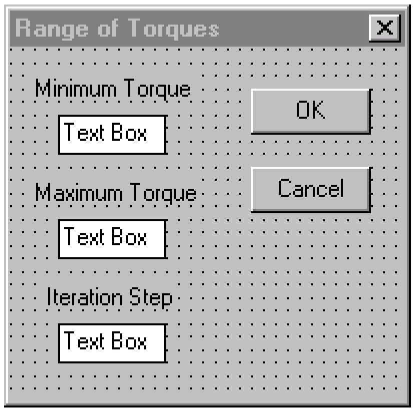


Figure 9-15
RangeOfTorque dialog box

20. **Select Save As... from the File menu and save the dialog box as RangeOfTorque.dlg.**

Using the Dialog Box in the Script

Once you have created a dialog box, you can use it in script files. To use a dialog box in a script:

1. **Click on the Range of Torque dialog box in the Dialog Editor.**

2. **Select Copy from the Edit menu.**

3. **Go to the Script Editor of Working Model.**

4. **Click on the line between the line Sub Main() and the line End Sub.**

5. **Select Paste from the Edit menu.**

   Notice that the visual representation of the dialog box in Dialog Editor is pasted into the Script Editor as text. The text contains information about the text objects and the text box objects in the dialog box.

Continuing the WM Script

1. **Beneath the line "End Dialog", press enter to insert a blank line and enter: Dim Doc as WMDocument, Driver as WMBody, DriverMotor as WMConstraint, Follower as WMBody, Measurements as WMOutput, InputDialog as UserDialog**

   This statement defines a variable called Doc of type WMDocument. Similarly, the variable Driver is defined as type WMBody, DriverMotor as type WMConstraint, Follower as type WMBody, Measurements as type WMOutput, and InputDialog as type UserDialog.

2. **Enter: Dim FName as String**

   The statement defines the variable, FName, of type String. It will hold the name of the file to which data will be output.

The type String is a standard BASIC type, whereas WMDocument, WMBody, WMConstraint, and WMOutput are types specifically defined for WM Basic.

3. **Enter: Set Doc = WM.ActiveDocument**

   A few lines above, we created a variable called Doc. Now we assign to it a value. This assignment tells WM Basic that Doc refers to the currently active document.

4. **Enter: Set Driver = Doc.Body("Driver")**

   This assignment tells WM Basic that Driver refers to the object we labeled Driver in the WM (Working Model) file. Note that a variable name in a WM script can be the same as the name of a body in the WM file.

5. **Enter: Set Follower = Doc.Body("Follower")**

   This assignment tells WM Basic that Follower refers to the object we labeled Follower in the WM file.

6. **Enter: Set DriverMotor = Doc.Constraint("Motor")**

   This assignments tells WM Basic that DriverMotor refers to the constraint we labeled Motor in the WM file.

7. **Enter: Dialog InputDialog**

   This makes InputDialog, the dialog box we created earlier, appear.

8. **Enter: Set Measurements = Doc.Output("Measurements")**

   This tells WM Basic that Measurements refers to the output meter we named Measurements.

9. **Enter: FName = SaveFileName$("FileName")**

   This statement calls up a dialog box that prompts you for a filename to which the Working Model data should be saved. The string, `FName`*, stores the filename you enter.*

10. **Enter: If FName = Empty then Exit Sub**

If the file you selected is empty (which only happens when you click "Cancel" in the dialog box), the script terminates.

11. **Enter: Open FName for Output as #1**

    This statement opens the file you selected in step 9. We will write our data to this file.

Now you will write a loop that tells Working Model to process the simulation. The user will specify in the dialog box a range of motor torques, by entering the minimum torque, the maximum torque, and the iteration step size. In addition, for each value of motor torque, you will write a sub-loop to tell to process the simulation for driver radius of 0.01 m to 0.05 m in steps of 0.01 m. Finally, for each value of driver radius, you will write another sub-loop to process the simulation for follower radius of 0.01 m to 0.05 m in steps of 0.01 m.

12. **Enter: For MotorTorque = InputDialog.MinimumTorque to InputDialog.MaximumTorque Step InputDialog.IterationStep**

    This creates a loop that starts counting from the minimum torque and goes up by the iteration step until it reaches the maximum torque. The variable MotorTorque is the "counter" that keeps track of the loop's progress. Note that the values MinimumTorque, MaximumTorque, and IterationStep are all information that the user entered in the InputDialog box earlier.

13. **Indent one tab and enter: For DriverRadius = .01 to .05 Step 0.01**

    This creates a loop that starts counting at 0.01 and goes up by 0.01 until it reaches 0.05. The variable DriverRadius is the "counter" that keeps track of the loop's progress.

14. **Indent one tab and enter: For FollowerRadius = .01 to .05 Step 0.01**

    This creates a loop that starts counting at 0.01 and goes up by 0.01 until it reaches 0.05. The variable FollowerRadius is the "counter" that keeps trach of the loop's progress.

15. **Enter: Doc.Reset**

    This resets the simulation.

16. **Enter: DriverMotor.Field.Value = MotorTorque**

    This sets the torque of the driver motor to MotorTorque, which will initially equal the minimum torque and go up by the iteration step until it reaches the maximum torque specified by the user.

17. **Enter: Driver.Radius.Value = DriverRadius**

    This sets the driver radius to DriverRadius, which will initially equal 0.01 and go up by 0.01 until it reaches 0.05.

18. **Enter: Follower.Radius.Value = FollowerRadius**

    This sets the follower radius to FollowerRadius, which will initially equal 0.01 and go up by 0.01 until it reaches 0.05.

19. **Enter: Doc.Run 5**

    This statement runs the simulation for 5 frames.

20. **Enter: BlockVelocity = Measurements.Column(4).Cell.Value**

    This assigns the value of the Measurements meter's fourth field, the BlockVelocity field, to the variable BlockVelocity shown here.

21. **Enter: ResultingTorque = MotorTorque * FollowerRadius / DriverRadius**

    This calculates the final torque available to lift the block as a result of the gear system.

22. **Enter: If BlockVelocity > 0 then**

    If the velocity block is greater than zero in the simulation, that is, if it is going up, we will output the data, as shown in the next step. Otherwise, we will ignore the data from that particular simulation.

23. **Indent one tab and enter: Print #1, "Driver: ";DriverRadius, "Follower: ";FollowerRadius, "Motor: ";MotorTorque, "Resulting Torque: ";ResultingTorque**

    This statement outputs the simulation data to the file created in the SaveFileName statement earlier. We refer to it here with the ID, #1 (assigned with the Open statement earlier).

24. **End if**

    This ends the If statement created in step 22.

25. **Next FollowerRadius**

    This increments the variable FollowerRadius by one step, in this case 0.01, and sends the program back to the start of the FollowerRadius for loop.

26. **Next DriverRadius**

    This increments the variable DriverRadius by one step, in this case 0.01 and sends the program back to the start of the DriverRadius for loop.

27. **Next MotorTorque**

    This increments the variable MotorTorque by one step, in this case the size of the IterationStep as specified by the user. It then sends the program back to the start of the MotorTorque for loop.

28. **Close #1**

    This closes the data file. Again we refer to the file by the ID, #1, that we gave it when we opened it.

29. **Doc.Reset**

    This resets the simulation.

30. **End Sub**

    This ends the script.

This script is finished, but you should save it before you run it.

31. **Choose Save from the File menu in the Script Editor window.**

    Name the file AdvancedScripting.wbs and click [OK].

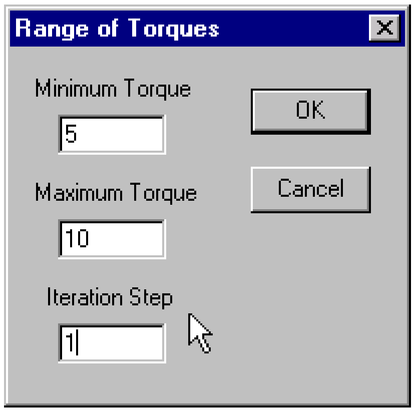
Running the Script

You are now ready to run the script.

1. **Choose Start from the Run menu in the Script Editor window or click the Start button in the Script Editor Toolbar.**

   The dialog box appears, Figure 9-16.

2. **Enter 5 for Minimum Torque, 10 for Maximum Torque, and 1 for Iteration Step.**



Range of Torques

Minimum Torque: 5

Maximum Torque: 10

Iteration Step: 1

Figure 9-16
Dialog Box

3. **Save output data to the file AdvancedScripting.txt in the same directory as AdvancedScripting.wbs.**

NOTE: The script created in this exercise is included as a file called **AdvancedScripting.WBS** in the Tutorial folder which was installed with Working Model. Users of 680x0-based MacOS systems (who cannot use the Script Editor) can still follow this exercise by running the script file.

To run the preinstalled script file:

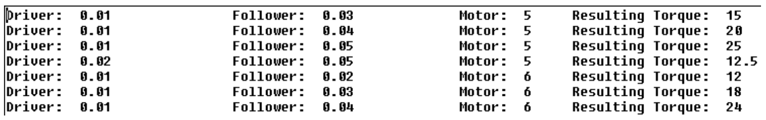
1. **Choose Run... from the Script menu.**

   A file browsing dialog appears.

2. **Open the Tutorials folder and select the file AdvancedScripting.WBS.**

3. **Click [Open].**

After you run your script, take a look at the output data file using a simple text editor such as Notepad (on Windows systems) or SimpleText (on MacOS systems). The output file should resemble Figure 9-17.



```
Driver:  0.01          Follower:  0.03          Motor:  5     Resulting Torque:  15
Driver:  0.01          Follower:  0.04          Motor:  5     Resulting Torque:  20
Driver:  0.01          Follower:  0.05          Motor:  5     Resulting Torque:  25
Driver:  0.02          Follower:  0.05          Motor:  5     Resulting Torque:  12.5
Driver:  0.01          Follower:  0.02          Motor:  6     Resulting Torque:  12
Driver:  0.01          Follower:  0.03          Motor:  6     Resulting Torque:  18
Driver:  0.01          Follower:  0.04          Motor:  6     Resulting Torque:  24
```

Figure 9-17
AdvancedScripting.txt file

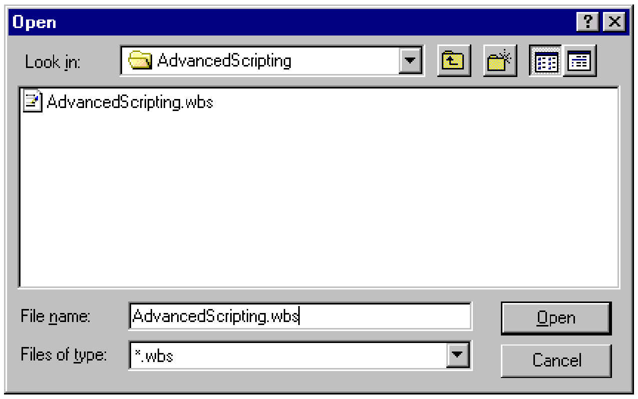
9.7. Adding a Script Button

Once you have written a script, it can be tedious to have to choose Run... from the Script menu and select the same script every time you want to run it. It is more convenient to be able to click a script button in the simulation.

To add a script button to your simulation:

1. **Select New Button under the Define menu and choose Script button... from the New Button submenu.**

   The Open window appear, Figure 9-18.



AdvancedScripting

AdvancedScripting.wbs

File name: AdvancedScripting.wbs

Files of type: *.wbs

Figure 9-18
Open window

2. **In the Open window, select the script you have created and click [Open].**

   A button labeled AdvancedScripting appears on the Working Model workspace.

3. **Click the AdvancedScripting button in the workspace to run the simulation.**