RECURDYN

汽车巡航教程（CoLink）

FunctionBay

Registered Trademarks of FunctionBay, Inc. or Subsidiary

RecurDyn™ is a registered trademark of FunctionBay, Inc.

RecurDyn™/SOLVER, *RecurDyn*™/MODELER, *RecurDyn*™/PROCESSNET, *RecurDyn*™/AUTODESIGN, *RecurDyn*™/COLINK, *RecurDyn*™/DURABILITY, *RecurDyn*™/FFLEX, *RecurDyn*™/RFLEX, *RecurDyn*™/RFLEXGEN, *RecurDyn*™/LINEAR, *RecurDyn*™/EHD(Styer), *RecurDyn*™/ECFD_EHD, *RecurDyn*™/CONTROL, *RecurDyn*™/MESHINTERFACE, *RecurDyn*™/PARTICLES, *RecurDyn*™/PARTICLEWORKS, *RecurDyn*™/ETEMPLATE, *RecurDyn*™/BEARING, *RecurDyn*™/SPRING, *RecurDyn*™/TIRE, *RecurDyn*™/TRACK_HM, *RecurDyn*™/TRACK_LM, *RecurDyn*™/CHAIN, *RecurDyn*™/MTT2D, *RecurDyn*™/MTT3D, *RecurDyn*™/BELT, *RecurDyn*™/R2R2D, *RecurDyn*™/HAT, *RecurDyn*™/曲柄, *RecurDyn*™/PISTON, *RecurDyn*™/VALVE, *RecurDyn*™/TIMINGCHAIN, *RecurDyn*™/ENGINE, *RecurDyn*™/GEAR are trademarks of FunctionBay, Inc.

Third-Party Trademarks

Windows and Windows NT are registered trademarks of Microsoft Corporation.

ProENGINEER and ProMECHANICA are registered trademarks of PTC Corp. Unigraphics and I-DEAS are registered trademark of UGS Corp. SolidWorks is a registered trademark of SolidWorks Corp. AutoCAD is a registered trademark of Autodesk, Inc.

CADAM and CATIA are registered trademark of Dassault Systems. FLEX*lm* is a registered trademark of GLOBEtrotter Software, Inc. All other brand or product names are trademarks or registered trademarks of their respective holders.

Edition Note

These documents describe the release information of ***RecurDyn™*** V9R1.

目录

Chapter 1

预备工作

目的

本教程使用 **CoLink** 仿真来控制动力学系统，**CoLink** 用于交互式环境设计、仿真和测试时变系统。在 **CoLink** 中定义控制系统去控制在 **RecurDyn** 中的机械系统。

本次仿真系统是简单的一维问题，该场景涉及到两辆车, 一辆慢车在前面, 一辆快车从后面追上来。设想开车时发现前面有车或突然有车进入车道，这种情况下可能会发生什么。控制器将自动调整应用于驱动轮的转矩使得两辆汽车距离处于安全值。用一个简单的 **PID** 控制器可以达到这个目的。

教程的最后是关于模型自适应巡航的复杂控制系统。

读者

本教程适用于已学习如何使用 **RecurDyn** 创建几何体、运动副和力元素的中级用户。前面教程有对其详细介绍。

预备知识

首先应该学习 3**D** 曲柄滑块机构和发动机与螺旋桨的教程, 或者有基本的物理知识。

步骤

本教程包括以下步骤，每个步骤所花时间如下表所示。

Procedures	Time（minutes）
打开初始模型	5
创建运动副和耦合器	15
定义新模型	15
集成 CoLink	15
增强 CoLink 模型	15
Total:	65

预计完成时间

大概需要 65 分钟完成。

Chapter

2

打开初始模型

任务目标

学习如何导入一个机械系统并完善模型，用于添加控制系统。

预计完成时间

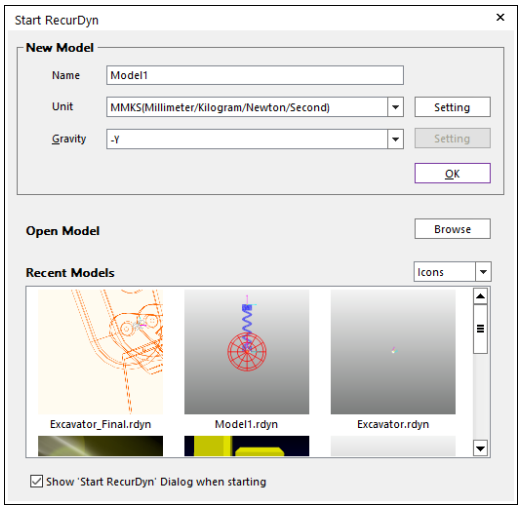
5 分钟

打开 RecurDyn

启动 **RecurDyn**，打开初始模型

1. 双击桌面上的 **RecurDyn** 图标。

2. 当 **Start RecurDyn** 对话框弹出时，关闭对话框，不需要创建新的模型，使用现有的模型。

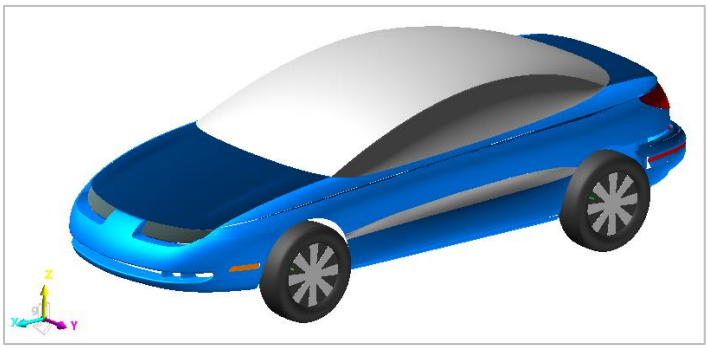


3. 在 **File** 菜单下，点击 **Open**。

4. 在 **CoLink** 教程目录下(<安装目录>**Help****Tutorial** **Colink****Tut2_Car**)，选择文件 **CarCruise_Initial. rdyn**。

5. 点击 **Open**。

模型如下图所示



保存初始模型

1. 在 **File** 菜单下，点击 **Save As**。

2. 将模型保存在不同目录下，因为不能在教程目录中仿真。

Chapter

3

创建运动副和耦合器

任务目标

本章创建几个轮子的运动副并将它们连接在向前运动的汽车上。定义汽车的初始速度和汽车运动时受到的滚动阻力和摩擦阻力，然后运行仿真。

预计完成时间

15 分钟

创建旋转副

本节创建连接车轮与车身的旋转副。

创建运动副

1. 将工作面改为 **XZ** 平面，可以使用快捷方式(**shift** + **A**)。

   保证旋转副方向的正确。

小贴士：按 **C** 在窗口中间显示汽车。

2. 按 **A** 打开自动操作模式去连续地执行同一任务。

小贴士:自动操作模式能够帮助重复执行相同的任务，不必多次从菜单中选择相同的任务。一旦打开自动操作，再从菜单中选择一个任务，**RecurDyn** 会重复完成这个任务,直到退出自动操作模式。

3. 在 **Professional** 标签下的 **Joint** 选项中，点击 **Revolute**。
4. 创建方式选择 **Body**，**Body**，**Point**。
5. 对于每一个轮子，在车身与轮子的中心坐标之间创建旋转副。
   - 点击 **Car_Body**。
   - 点击 **Wheel**。
   - 点击轮子的中心坐标。

   重复以上步骤完成其他轮子旋转副的建立。
6. 完成旋转副的创建后，按 **A** 关闭自动操作模式，然后按 **ESC** 取消旋转副的创建。
7. 重命名新创建的运动副，例如 **Rev_Front_Right**，**Rev_Front_Left**, **Rev_Rear_Right**, **Rev_Rear_Left**。

小贴士:在数据库窗口中，右键点击一个旋转副，点击 **Property**，更改名字。

模型如下图所示

创建平动副

创建平动副来定义汽车运动。

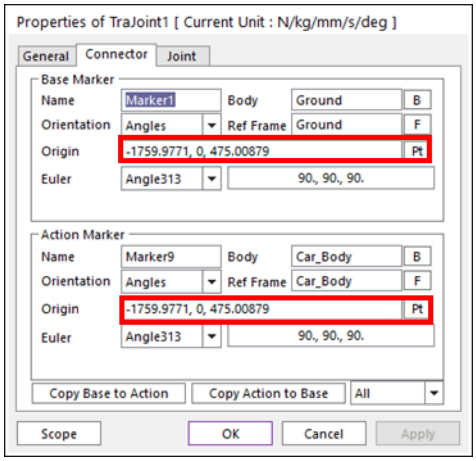
创建运动副：

1. 在 **Professional** 标签下的 **Joint** 选项中，点击 **Translate**。

2. 使用 **Body, Body, Point, Direction** 创建方式，选择 **Ground** 及 **Car_Body**，输入下面 **Point** 坐标。

   - **−1759. 9771, 0, 475. 00879**

3. 点击 **Ground Inertia Marker** 中的**+X** 轴。



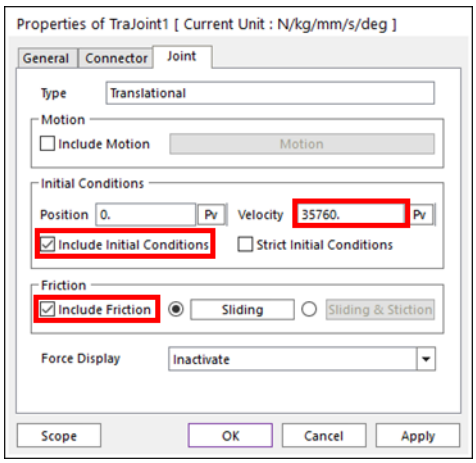
4. 改变平动副的属性(右键点击运动副，选择 **Property**)。

   - 在 **Properties** 对话框的 **General** 标签中，重命名平动副为 **Tra_Car_Ground**。

   - 在 **Joint** 页面，为运动副设置初始速度。

     a. 勾选 **Include Initial Conditions**。

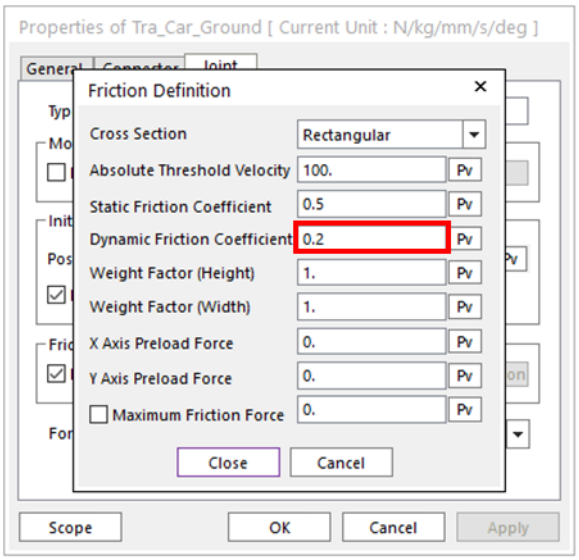
     b. **Velocity** 设置为 **35760mm/s**(35. 76**m/s** 或者 80**mph**)。



设置模型摩擦属性，仿真汽车的阻力和摩擦力。

5. 点击 **Include Friction**。

6. 点击 **Sliding** 并将 **Dynamic Friction Coefficient** 改为 **0. 2**，如下图所示。



7. 点击 **Close**，再点击 **OK** 保存更改并退出。

现在，已经为创建车轮与前进的汽车之间的耦合器做好了准备。

创建耦合器

创建耦合器，使轮胎角速度与汽车速度相匹配。

创建耦合器：

1. 在 **Professional** 标签下的 **Joint** 选项中，点击 **Coupler**。
2. 使用 **Joint**，**Joint**，**Joint** 创建方式，点击 **Tra_Car_Ground**，然后点击 **Rev_Front_Right** 最后点击 **Rev_Front_Left**。
3. 右键点击新创建的耦合器来显示其属性对话框。计算耦合器的比例因子。

小贴士：计算耦合器的比例因子

耦合器规模有下面方程定义，**d** 表示耦合器的比例因子，**u** 代表运动副的速度。

$$d_1 \times u_1 + d_2 \times u_2 + d_3 \times u_3 = 0$$

汽车前进速度与轮胎的转速之间的基本关系为 $v = w \times r$，
其中

- **v** 为汽车速度
- **w** 轮子的转动角速度
- **r** 轮子的半径

当然，这里假设汽车只是直线前进，所以两个车轮都以相同的速度旋转。这一约束将在下一步骤中由另外一个耦合器执行。车轮半径是 336 毫米，所以上面的方程变为如下形式：

$$d_1 \cdot r \times w + d_2 \times w + d_3 \times w = 0$$
$$d_1 \cdot r + d_2 + d_3 = 0$$

因为轮子都有相同的半径，所以 **d**2=**d**3。假设 **d**1 = 1，因为比例因素是相对的，需要确定其中的一个。这就形成以下方程：

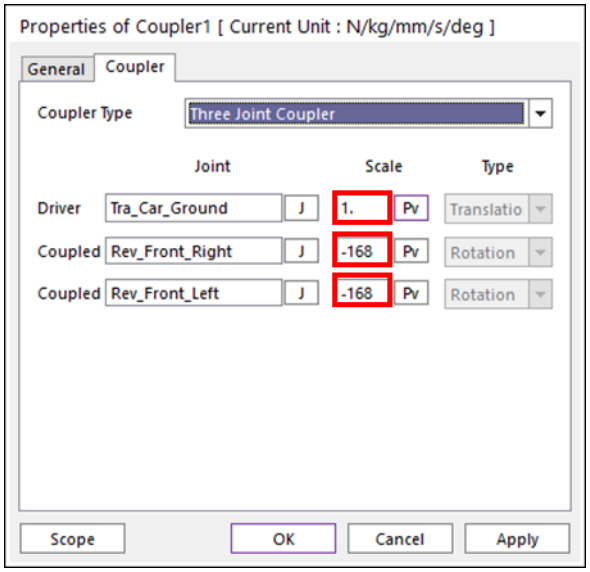
$$r + d_3 + d_3 = 0$$
$$d_3 = -r/2$$
$$d_3 = -336/2$$
$$d_3 = -168$$
$$d_2 = -168$$

总之，平动副的比例因子是 1，两个旋转副的比例因子为**−168**。

4. 在 **Coupler Properties** 对话框中的 **Coupler** 选项里，输入比例因子 **1**，**−168** 和**−168**。

5. 在 **General** 页面中，重命名为 **Coupler_Front**，点击 **OK**。

6. 重复 1 到 5 步，设置后轮，重命名为 **Coupler_Rear**，并使用相同的比例因子。



如前所述，现在将在两个前轮之间创建一个耦合器，保证它们之间的旋转角速度相同。

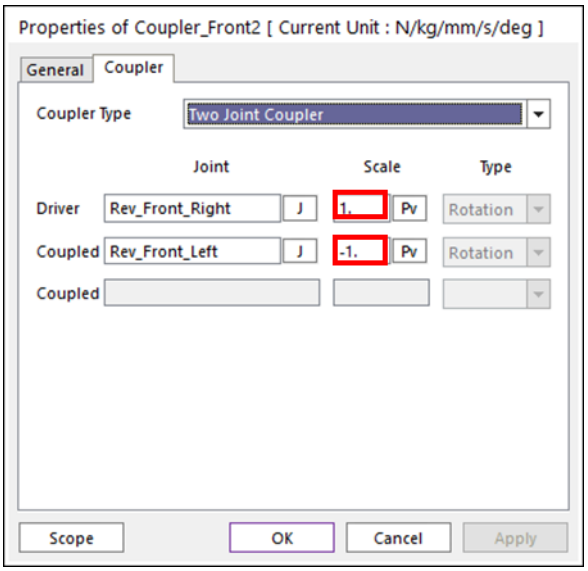
创建耦合器

1. 在 **Professional** 标签的 **Jiont** 选项中，点击 **Coupler**。

2. 使用 **Joint**，**Joint** 创建方式，点击 **Rev_Front_Right**，然后点击 **Rev_Front_Left**。

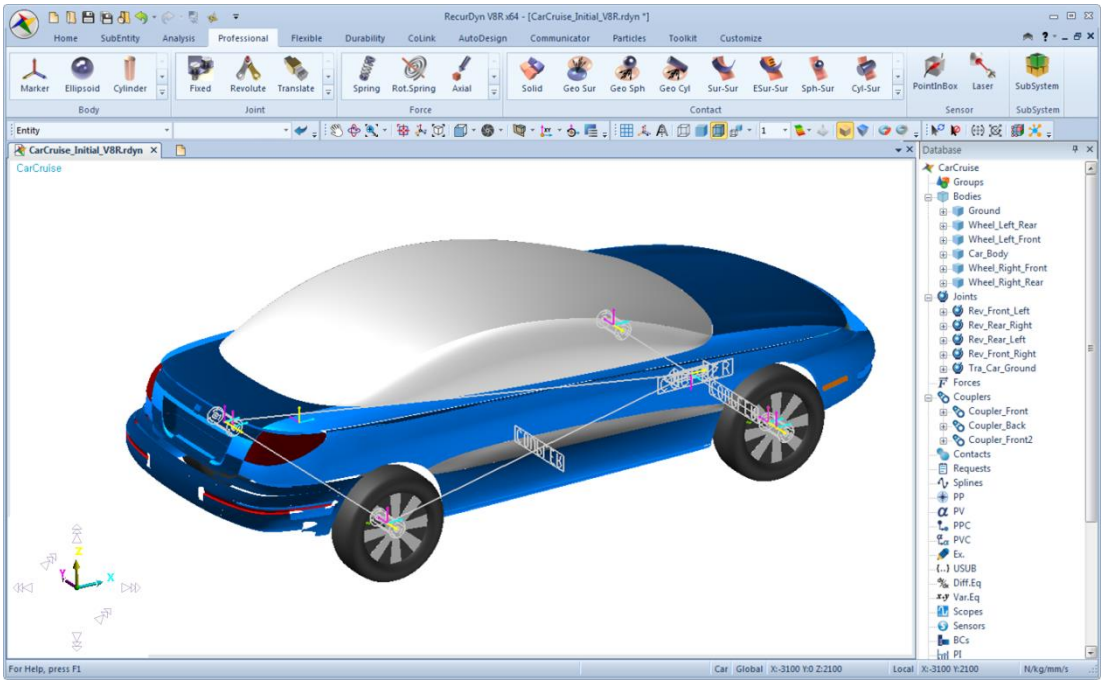
3. 编辑建立的耦合器的属性，做出如下更改：

   - 重命名为 **Coupler_Front2**。
   - 比例因子为 **1** 和 **−1**，本质上这两个转速必须相等。
   - 点击 **OK**。



4. 对左右后轮，重复 1 到 3 步骤，重命名为 **Coupler_Rear2**，使用相同的比例因子（**1** 和 **−1**）。

完成这些步骤后，模型如下图所示。

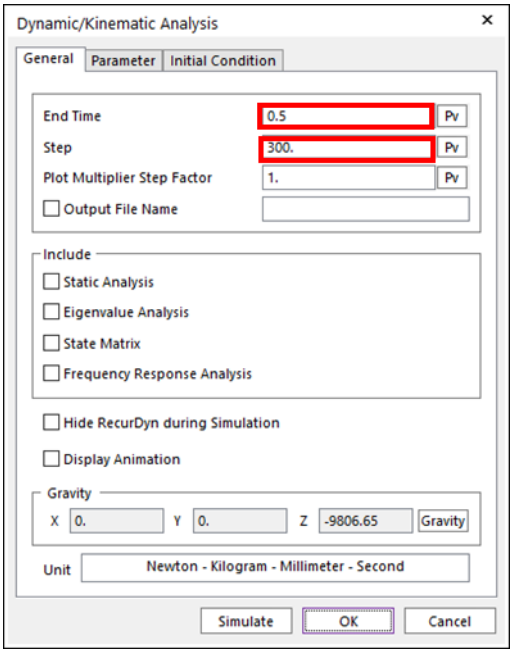


运行仿真

准备运行仿真。

运行仿真：

1. 在 **Analysis** 标签下的 **Simulation Type** 选项中，点击 **Dyn/Kin**。
2. 设置仿真，运行时间 **0.5** 秒，**300** 步，如右图所示。

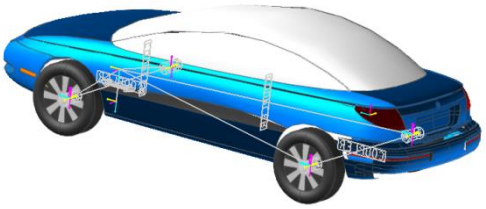


这将提供足够的时间来观察车轮和车的运动。

3. 点击 **Parameter** 选项，将 **Maximum Time Step** 设置为 **1. e−003**。
4. 点击 **Simulate**。

查看结果

查看结果



- 在 **Analysis** 标签的 **Animation Control** 选项中，点击 **Play** 键。

小贴士：如果汽车没有动怎么办？
仔细检查所有设置的耦合器系数的大小和符号。例如，这两个系数若设置为+1，汽车不会移动。

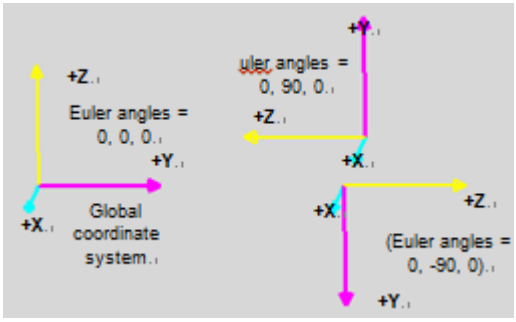
由仿真结果可知，这个设置有缺陷。这取决于如何设置旋转副，当汽车驱动前进时一些轮子可能在错误的方向旋转。解决方案：下一步将确保 **z** 轴的旋转副指向同一个方向。

修复旋转副

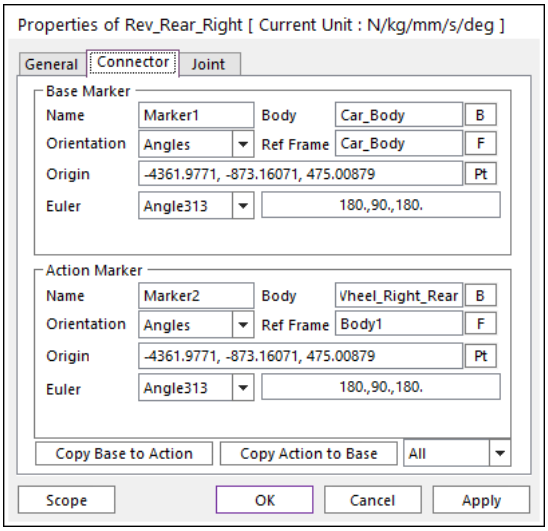
修复旋转副

1. 对于任何按错误方向旋转的车轮的旋转副，在数据库窗口中，右键点击运动副名称，然后点击属性对话框来显示其属性。

   下面以 **Rev_Rear_Right** 运动副为例。



2. 将 **Base Markers** 和 **Action Markers** 的 313**Euler** 角度从 **0, 90, 0** 改为 **0, −90, 0**，点击 **Apply**。



注：RecurDyn 将欧拉角从 **0, −90, 0** 改为 **180, 90, 180。**这没有问题，因为两个组合的欧拉角位置表征同一方向。（参见下一页关于欧拉角的附加信息的提示）。这改变的方向标记和对话框如右图所示，点击 **OK**。

3. 点击 **OK**。

4. 对于其他需要更改的运动副，重复上述步骤。

5. 重新运行仿真，检查问题是否被解决。

此时，小车模型创建完成。接下来，复制汽车模型并建立模型与 **CoLink** 的集成。

小贴士：欧拉角作用原理

在 **RecurDyn** 中，标准的欧拉角是：3−1−3**Eulerangles**。欧拉角如下步骤旋转得到

- 首先，绕#3 旋转（**or** +**Z**）轴。
- 然后，绕新位置的#1 旋转（**or** +**X**）轴。
- 最后，绕着新位置的#3 旋转（**or** +**Z**）轴。

右图的坐标系是空间坐标系， +**X** 轴正向旋转 90 度形成右上方的坐标系，+**X** 轴负向旋转 90 度形成右下方的坐标系。注意，欧拉角从 0，90，0 转到 0，−90, 0 时 **Z** 轴是逆时针转动的。

Chapter

4

定义新的模型

任务目标

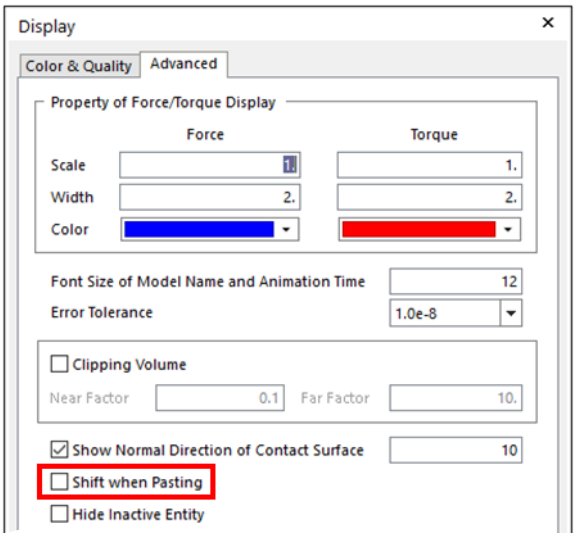
本章通过复制第一辆车创建第二辆新车模型。移动第二辆车，改变它的颜色，并使之以较慢的恒定速度行驶。

预计完成时间

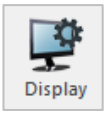
15 分钟

创建第二辆车

复制第一辆车，移动并修改属性。



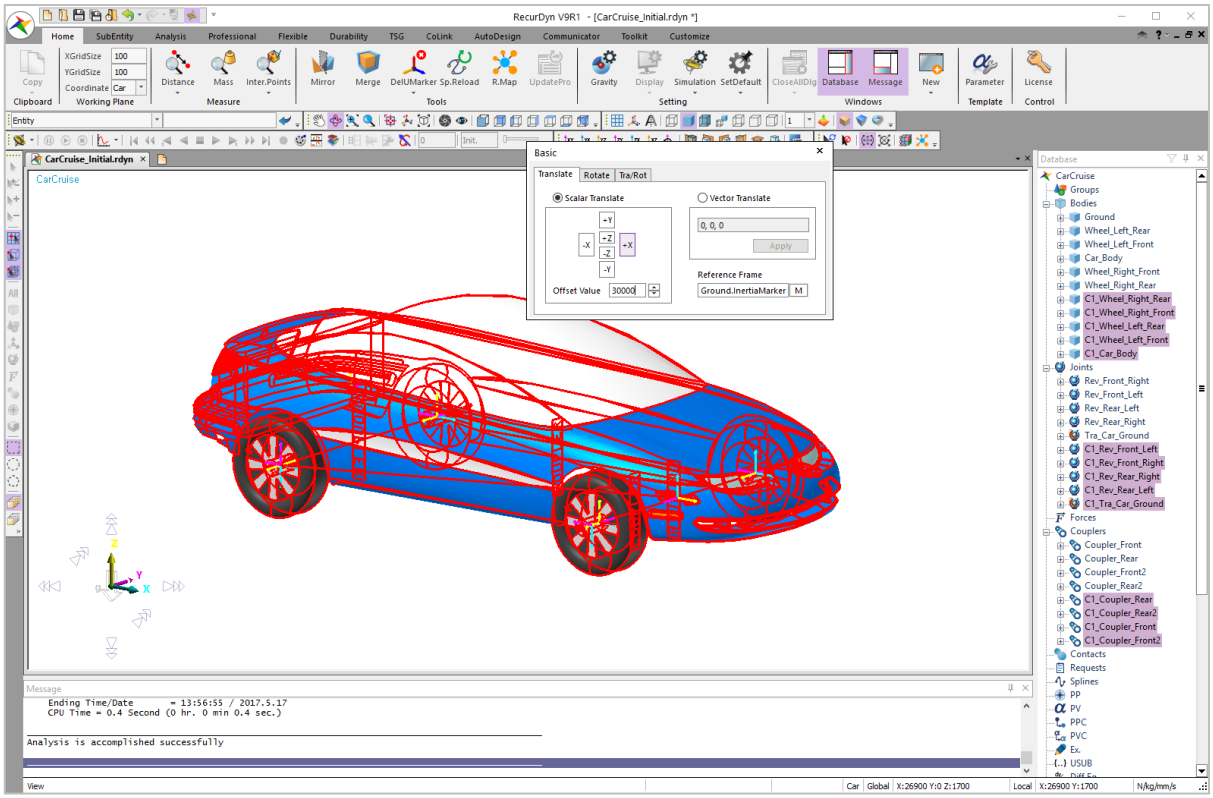
复制汽车模型



1. 在 **Home** 标签下的 **Model Setting** 选项里，点击 **Display**，取消 **Advanced** 中的 **Shift When Pasting** 的勾选，如右图所示。
2. 点击 **OK**。
3. 左键框选整个汽车。
4. 按 **Ctrl-C** 复制，按 **Ctrl-V** 粘贴一个相同的汽车模型。

小贴士:也可以点击 **Home** 标签的 **Clipboard** 选项里的 **Copy**，然后点击 **Clipboard** 选项里的 **Paste**。

5. 仍然选中复制的汽车模型，从工具栏，点击 **Object Control** 工具。
6. **+X** 方向移动复制的汽车模型 **30000mm**，如下图所示。



修改第二辆车

修改第二辆车

1. 打开第二辆车的平动副的属性对话框，在数据库窗口，右键 **C1_Tra_Car_Ground**，点击 **Property**。
2. 因为第二辆车以恒定速度运行，取消勾选 **IncludeFriction**。
3. 点击 **OK**。
4. 需要降低汽车的行驶速度，将 **C1_Car_Body** 的初始速度改为 **22, 350mm/s**，即 **50mph**。

小贴士：改变 C1_Car_Body 的初始速度

1. 在数据库窗口，打开 **C1_Car_ Body** 的 **Property** 窗口。
2. 选择 **Body** 项。
3. 点击 **InitialVelocity** 键。
4. 点击 **X** 方向的 **Translational Velocity**。
5. 输入 **22350**。
6. 定义 **Translational Velocity** 的 **Reference Marker** 为 **Ground. InertiaMarker**。
7. 定义 **Rotational Velocity** 的 **Reference Marker** 为 **C1_Car_Body. CM**。
8. 点击 **Close**。
9. 点击 **OK**。

---

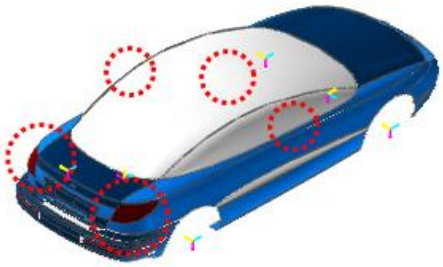
5. 下面，改变汽车的颜色使其更适合在高速公路上慢慢行驶。
   - 双击 **C1_Car_Body**，打开 **Body Edit**。

小贴士：可以右键 **C1_Car_Body**，点击 **Edit** 进入 **Body Edit** 界面。

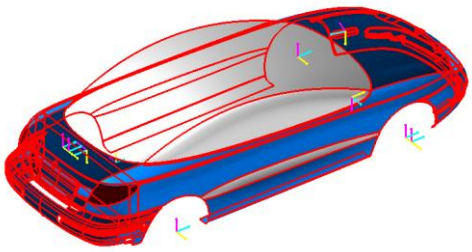
   - 在整个车身周围画一个选择框。然后, 按住 **Ctrl** 键, 点击图中三个白色的部分来清除它们，即车顶、窗户以及两个红色尾灯。

小贴士：旋转汽车模型以便看到汽车后轮，或在数据库窗口清除下面几个选项：

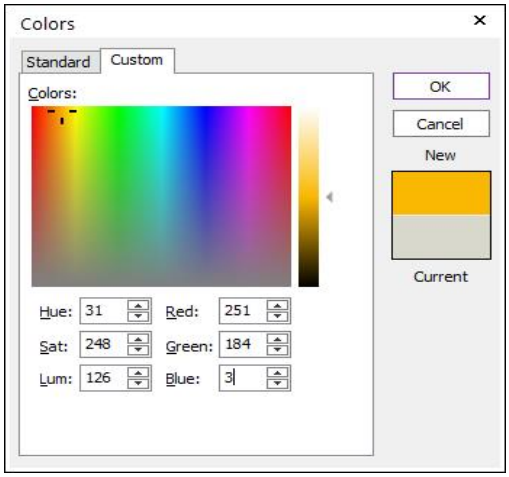
- C1_C1_C2_ImportSurface1
- C1_C1_C3_ImportSurface1
- C1_C3_ImportService1
- C1_C2_C1_ImportSurface1
- C1_C2_C2_ImportSurface1



模型如下图所示：



- 右键点击的物体上任何位置，点击 **Property**。
- 点击 **Graphic Property** 选项，将 **Color** 改为其他不同的颜色。
- 在 **Colors** 对话窗口中的 **Custom** 选项中，分别将 **Red**，**Green** 和 **Blue** 的数值改为 **251**，**184** 和 **3**。
- 两次点击 **OK**，接受并应用颜色更改



根据计算机的速度，可能需要等待几分钟。更改生效后，点击 **OK**。

6. 点击 **Exit** 退出 **Body Edit** 模式回到原模型建立界面。

模型已准备好与 **CoLink.** 进行集成。

Chapter

5

集成 CoLink

任务目标

本章设置模型与 **CoLink** 集成，并创建 **CoLink** 模型。将系统的仿真结果绘图。

预计完成时间

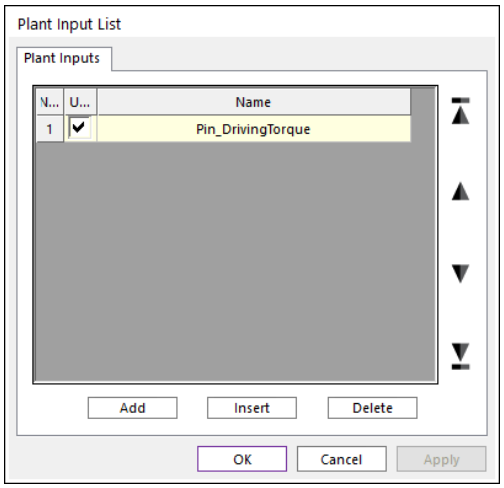
15 分钟

创建 Plant Input

首先创建控制系统对模型的输入，该元素创建为一个占位符，随后会对其定义。

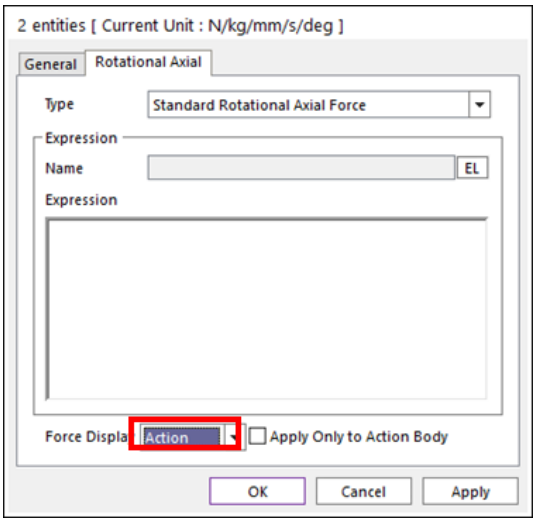
创建 Plant Input：

1. 在 Colink 标签下的 CoLink 中，点击 Plant Input。
2. 点击 Add，将 Plant Input 重命名为 Pin_DrivingTorque。
3. 点击 OK。



创建扭矩：

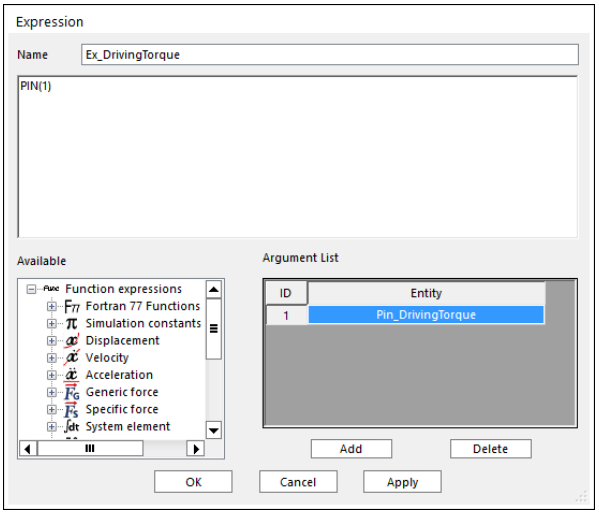
1. 在 Professional 页面的 Force 中，点击 Rotational Axial。
2. 将 Creation Method 设置为 Joint，选择 Rev_Front_Right。
3. 重复上述步骤对 Rev_Front_Left 进行设置。
4. 点击 RotationalAxial1，然后按住 Ctrl，点击 RotationalAxial2。这样两个就都被选中了，右键它们，选择 Property。
5. 为了在动画中显示计算出的扭矩，设置 Force Display，在对话框的底部，设置为 Action。
6. 点击 EL，输入扭矩计算表达式。



7. 在表达式列表对话框中，点击 **Create**。

8. 将表达式命名为 **Ex_DrivingTorque**，表达式为 **PIN(1)**。

9. 在 **Argument List** 中，点击 **Add**，然后在数据库窗口中，将 **Pin_DrivingTorque** 元素拖入框内，如右图所示。



小贴士：或者，可以双击黄色框输入 **Pin_DrivingTorque**。

10. 点击 **OK**，完成所有的更改。

创建 Plant Output

从控制系统模型输出两辆车的相对速度和它们之间的距离。

创建 **Plant Output**

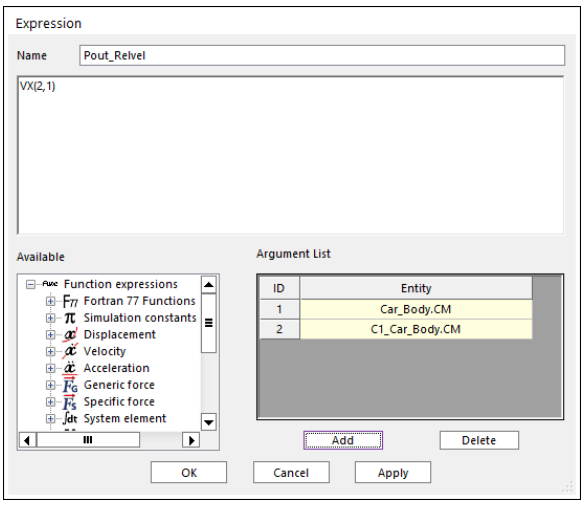
1. 在 **Colink** 标签下的 **Colink** 中，点击 **Plant Output**。

2. 在 Plant Output 的对话框中，点击 **Add**。

3. 在 Expression 对话框中，将表达式重命名为 **Pout_RelVel**，表达式为 **VX(2, 1)**。

4. 在 **Argument List** 中增加两个实体，在黄色窗口中输入 **Car_Body** 和 **C1_Car_Body**，如右图所示。

5. 点击 **OK**，完成更改，并返回到 Plant Output 列表。



6. 在 **Plant Output List** 中，再次点击 **Add**，重复 3 到 5 的步骤，将表达式由 **VX** 换为 **DX**，并重命名为 **Pout_Distance**。

7. 点击 **OK**。

创建 CoLink 模型

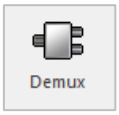
打开 **CoLink** 并创建控制系统，包括创建方框图，包含 **RecurDyn** 模型模块、**PID** 控制器模块、求和模块和一个恒定参考信号。

创建 **CoLink** 模型

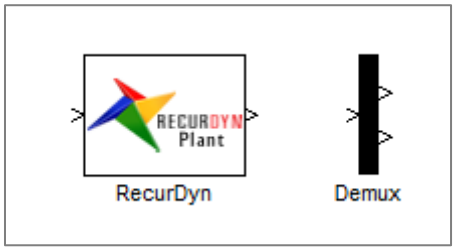
1. 在 **Colink** 标签下的 **Colink** 中，点击 **Run CoLink**。

   CoLink 会打开，但没有任何模型。

2. 在 **Connector** 标签下的 **Link** 中，点击 **RecurDyn** 块，创建方框图，如下图所示。



3. 在 **Connector** 标签下，点击 **Demux** 方块，并将其放在 **RecurDyn** 方块的右边，如下图所示。

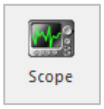


   这将 **RecurDyn block** 的单一输出分解为两个输出：**RelVel**（相对速度）和 **Distance**（相对距离）。

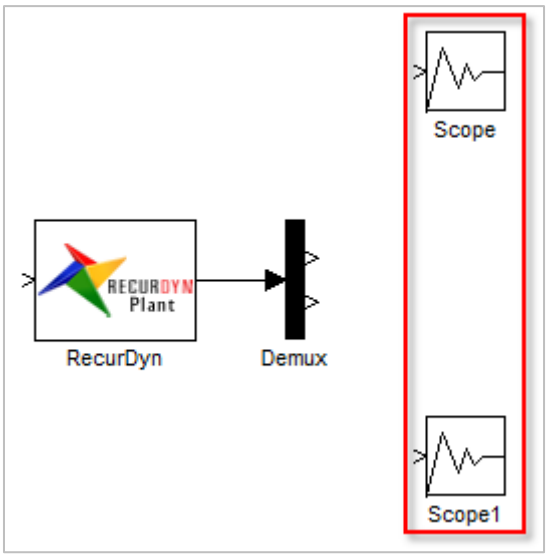
4. 为了将两个方框图连接在一起，点击 **RecurDyn** 块，按住 **Ctrl** 键，并点击 **Demux** 块。

   这是连接方框图的方法。在其余 **CoLink** 模型的创建中，使用相同的方法连接。

若需要观察仿真的结果，可以创建相对速度和相对距离信号示波器，示波器显示 **x**–**y** 图，其中 **x** 为时间，**y** 为输入数据。



5. 在 **General** 标签下的 **Output** 中，点击 **Scope** 块，在模型窗口中创建两次，将一个示波器放在右上方，一个示波器放在右下方，如图所示。

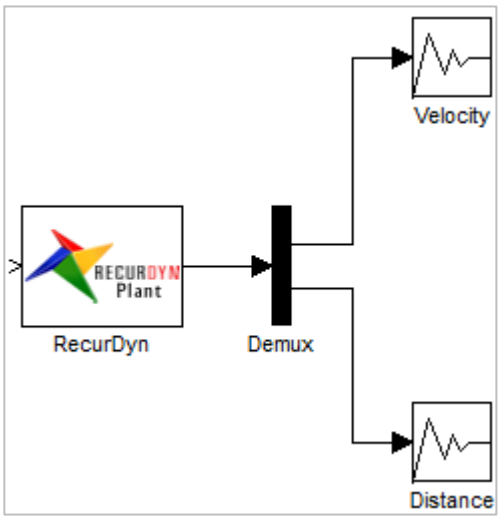


6. 用相同连接方法将 **Demux** 块与两个 **scopes** 连接。

7. 双击右上方的 **scopes**，并重命名为 **Velocity**。将右下方的 **scope** 重命名为 **Distance**。

8. 保存模型，命名为 **CarCruise_Control**。

模型如下图所示：

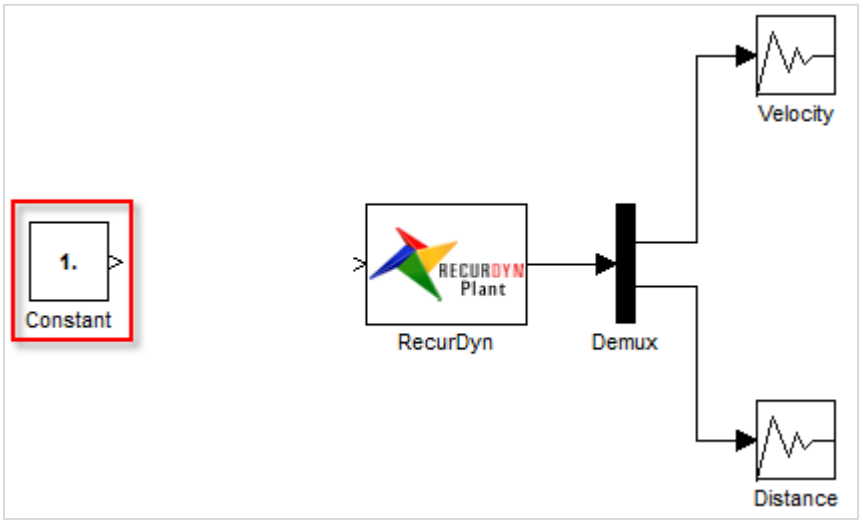


创建比例反馈控制

下面，创建一个参考输入信号和反馈回路，添加比例控制。这是构建一个 **PID** 控制器的第一步。

创建比例反馈控制

1. 在 **General** 标签下 **source** 中，点击 **Constant** 块，将其放在离中间较远的左边，如下图红色方框所示。

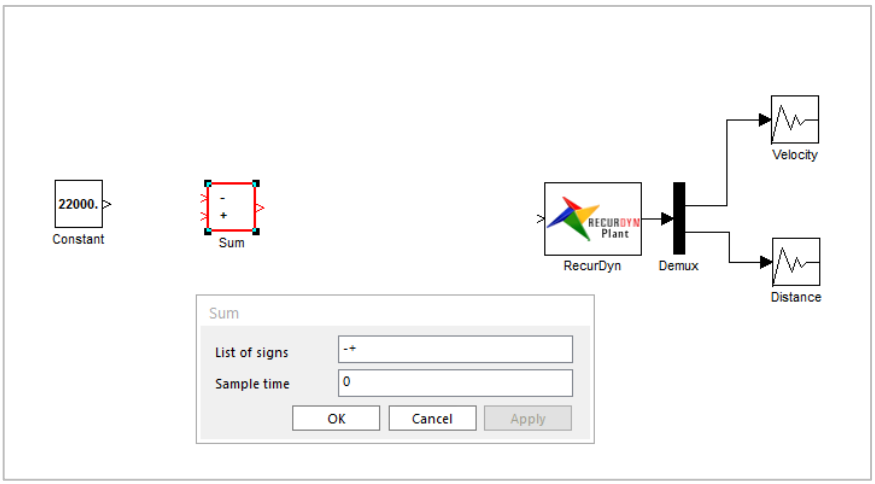


这个常数表示两个汽车之间的安全距离。

2. 双击 **Constant** 块，并设为 **22, 000**。

3. 在 **Math** 标签中，点击 **Sum** 块，在模型窗口中创建，并将其放在 **Constant** 块的右边。

4. 将 **Constant** 块与 **Sum** 块的上部连接，双击 **Sum** 块，将 **list of signs** 改为-+，如下图所示。



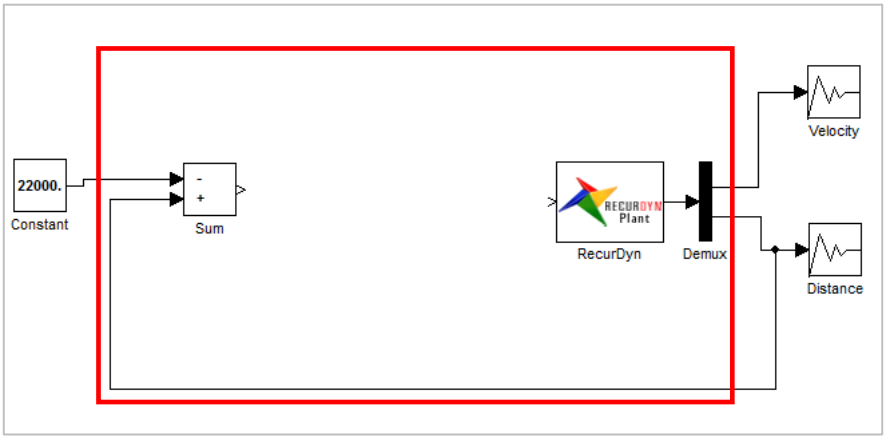
5. 点击 **OK**。

现在将 **Distance** 信号与 **Sum** 块下部连接，这将产生一个距离反馈信号。

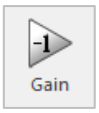
6. 将 **Distance** 的输出与 **Sum** 块的下部连接，如下图所示：

- 点击 **Demux** 块与 **Distance Scope** 之间的连线来选中它。
- 右键选中连线，拖住鼠标直至 **Sum** 块的底部，然后释放鼠标右键。

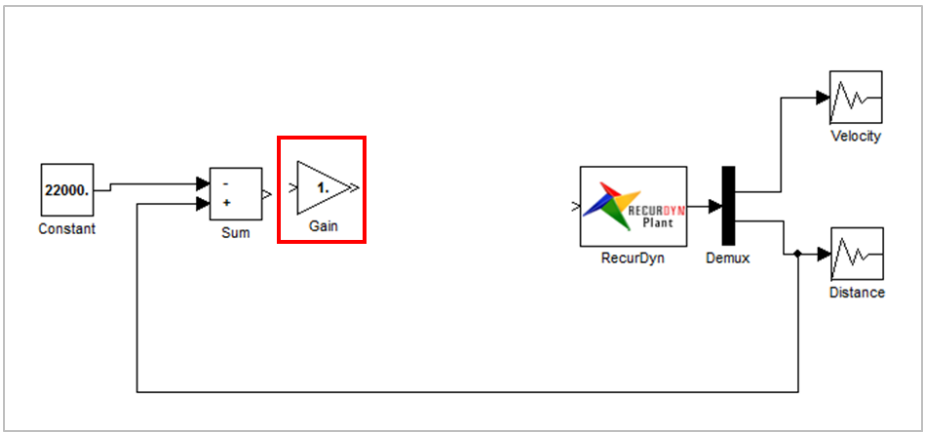
这会在右键单击的地方创建一个连接。



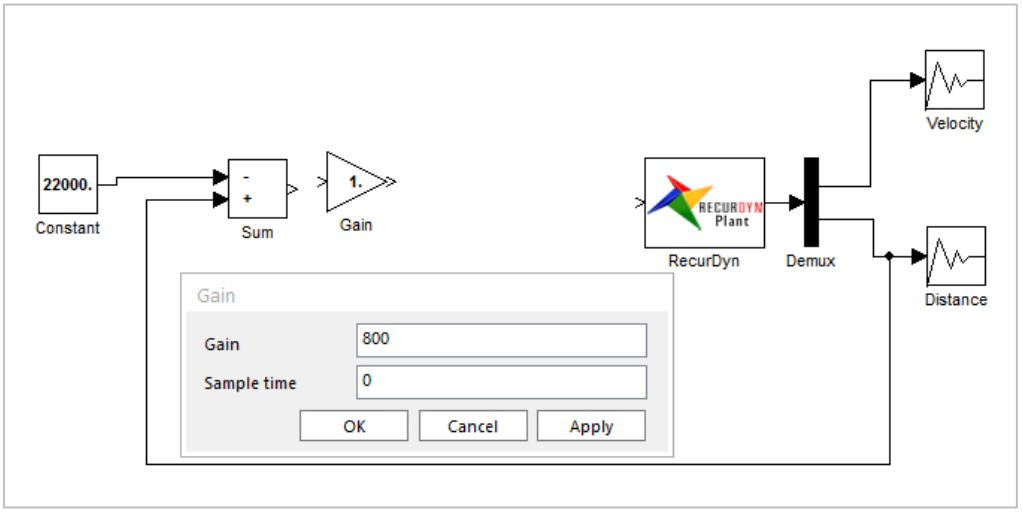
模型应如上图所示



7. 在 **Math** 标签下，点击 **Gain** 块，在模型窗口中创建块，并将其放在 **Sum** 块的右边，如下图所示。



8. 将 **gain** 改为 **800**，并将两块连接，如上图所示。



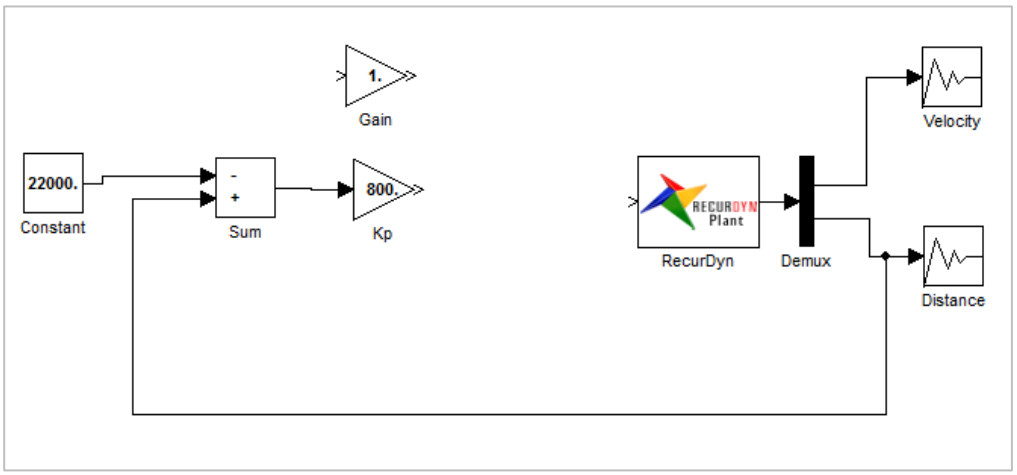
9. 将 **Gain** 块重命名为 **Kp**，它是 **PID** 控制器的比例控制增益。

添加微分与积分控制

添加微分和积分反馈控制器。因为距离信号的导数是速度，使用速度直接作为输入微分增益。对于积分控制，需要添加一个积分器模块和作为输入的积分增益。

添加微分控制

1. 在 **Math** 标签下，点击 **Gain** 块，在模型窗口中创建，并将其放在 **Kp** 块的上方，如下图所示：



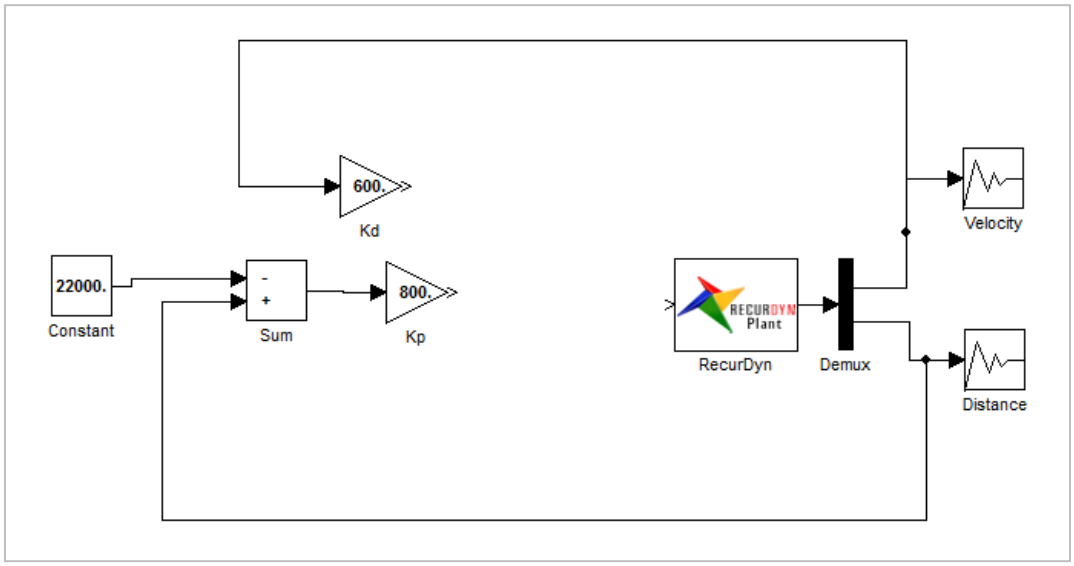
2. 将其重命名为 **Kd**，作为微分控制增益。

3. 将 **Velocity** 输出与 **Gain** 块连接，如下图所示。

   - 点击 **Demux** 块与 **Velocity Scope** 之间的连线，完成选中。
   - 右键选中连线的转折点，拖住鼠标直至 **Gain** 块的底部，然后释放鼠标右键。

4. 将 **gain** 值改为 **600**。

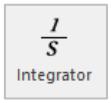
模型如下图所示：



添加积分控制

1. 在 **Math** 标签下，点击 **Gain** 块，在模型窗口中创建，并将其放在 **Kp** 块的下方，如下图红色区域所示。

2. 将块重命名为 **Ki**，并改变其数值为 **100**。

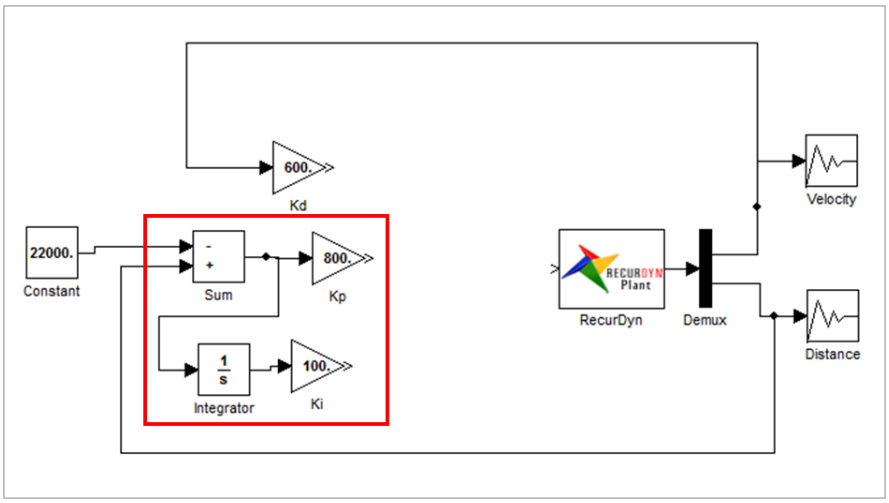


3. 在 **Continuous and Discrete** 标签下的 **Continuous** 块中，点击 **Integrator** 块。

4. 右键点击 **Sum** 块与 **Kp** 块的连接，并将其信号同时供给 **Integrator** 块。

5. 将 **Integrator** 块与 **Ki** 块连接。

模型如下图所示。



形成控制回路

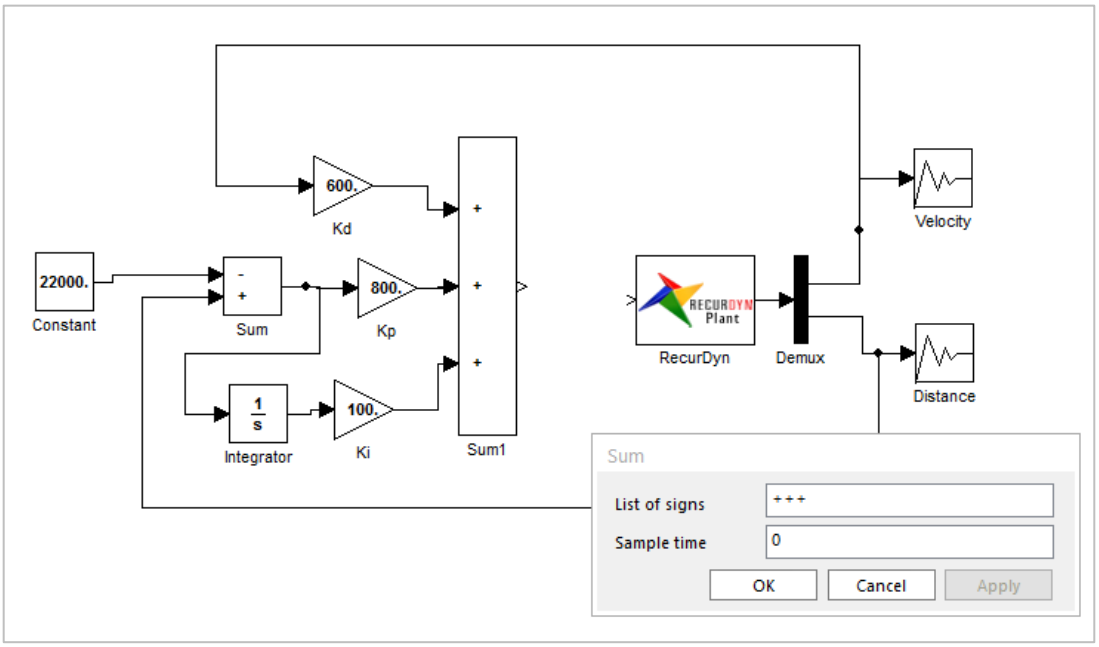
下面，三个反馈信号相加，并与 **RecurDyn** 模块连接，形成回路。

形成回路

1. 在 **Math** 标签下，点击 **Sum** 块，在模型窗口中创建，并将其放在 **Gain** 块与 **RecurDyn** 块之间，如下图所示。

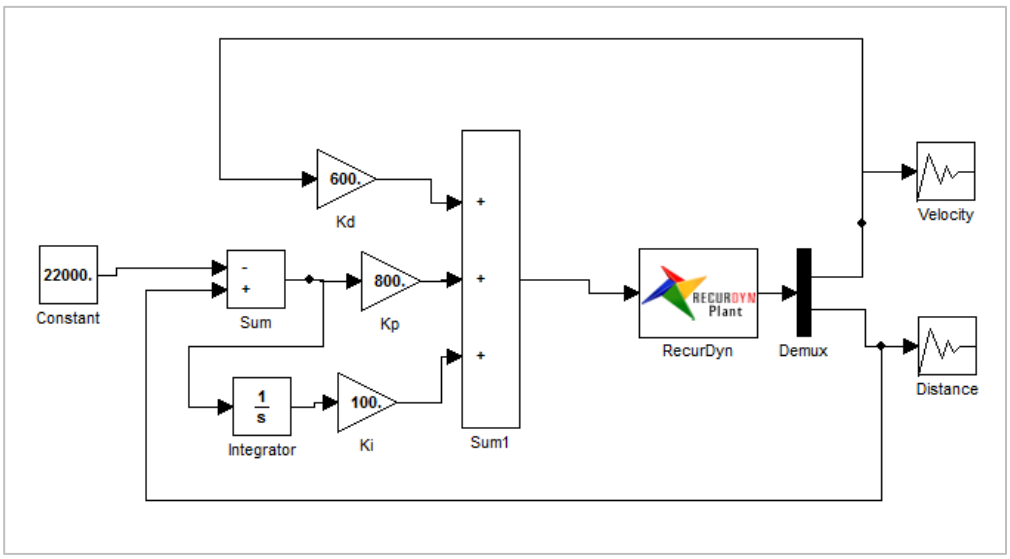
2. 双击新添加的 **Sum** 块，将 **list of signs** 改为+++。

   将三个反馈信号累加成一个输入信号。



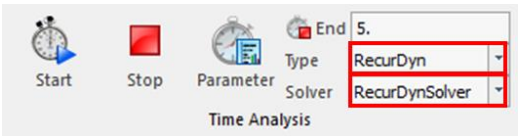
3. 将 **Gain** 块与 **Summer** 块连接，同时，**Summer** 块与 **RecurDyn** 块连接。

   已经完成了控制器的创建，如下图所示。



仿真模型

仿真模型并产看结果。



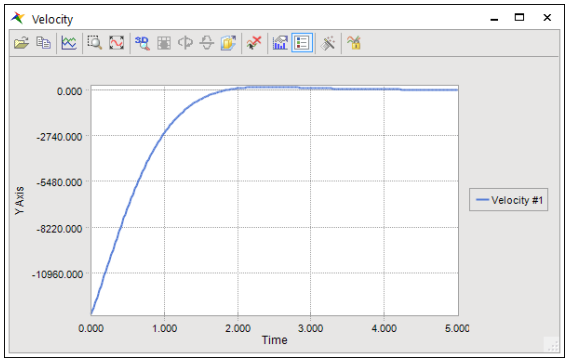
仿真模型：

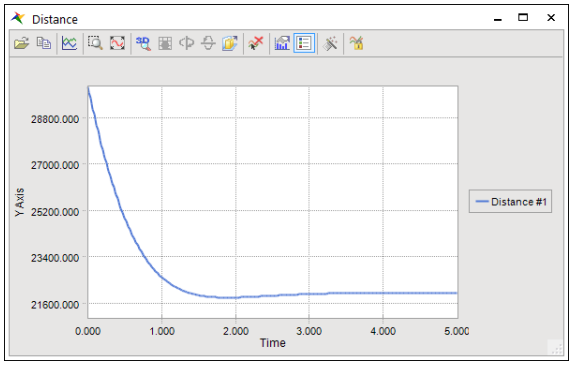
1. 将 **Type** 改为 **RecurDyn**，将 **Solver** 改为 **RecurDyn Solver**。

2. 在 **Simulation** 标签下的 **Time Analysis** 组中，点击 **Start**。

3. 当仿真完成后，双击 **Velocity** 和 **Distance scopes**，并查看结果。结果如右图所示。

   速度收敛于零，表明瞬态响应后，两辆车保持一个固定的车距。右图显示的距离最终值是 22000 毫米，如预期相同。响应稳定有点慢，过度时间太长。下面解决这些，再次运行仿真。





小贴士：如果联合仿真不运行怎么办?
如果联合仿真不运行, 服务器繁忙的对话框将会弹出。如果对话框出现：

- 点击 **Switchto**，在 **RecurDyn** 输出窗口中察看信息

如果 **CoLink** 模型不能发现错误消息, 检查 **RecurDyn** 和 **CoLink** 模型的位置。确保两个模型都位于相同的目录中。如果不是, 将两种模型放在同一目录中并重新启动 **RecurDyn** 和 **CoLink**。如此仿真应该能正常进行。

调整控制增益：

1. 双击 **Kp** 块将 **gain** 改为 **1000**，另外，将 **Kd** 的改为 **800**。
2. 点击 **Start**，运行仿真

   注意，示波器的图形没有改变。需要再次运行仿真，更新图形结果。

   此时，系统动力学行为是合理的。接下来，回到 **RecurDyn** 查看动画和图像结果。

查看结果

本部分绘制扭矩、两辆车之间的相对速度和相对距离图形，并输出动画。

查看结果：

1. 回到 **RecurDyn** 程序，保持 **CoLink** 程序打开，方便对模型进行拓展。
2. 

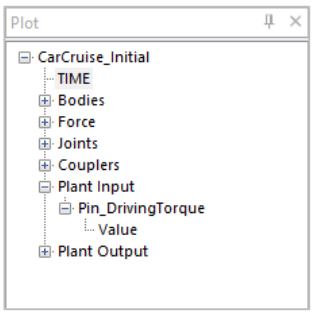
打开 **Plot window**。
3. 

在工具栏中的工具里，点击 **Show All Windows** 使得窗口分成 4 部分。
4. 点击左上方的分窗口显示。
5. 

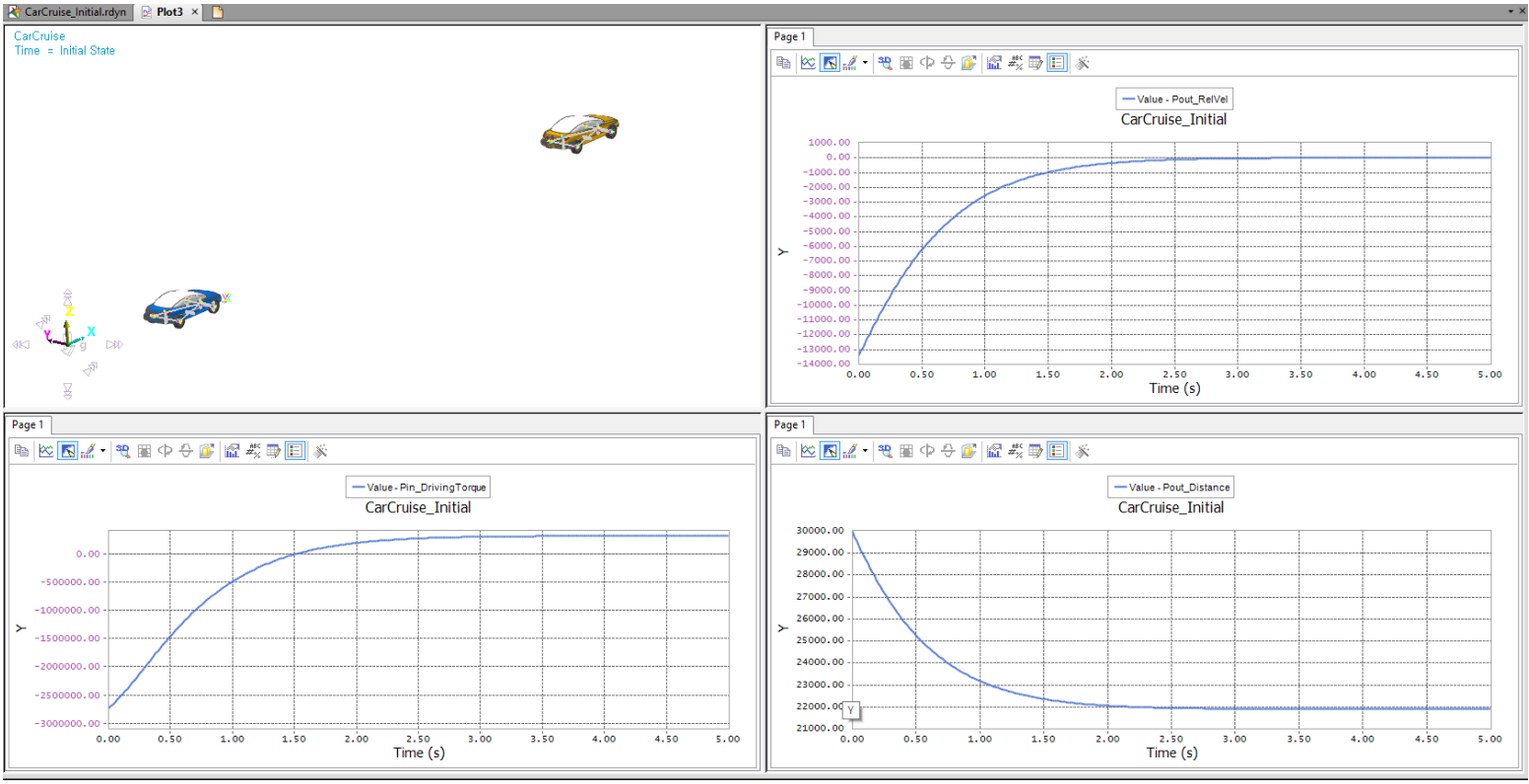
在 **Tool** 标签下的 **Animation** 中，点击 **Load Animation**，加载动画。

   弹出关于覆盖分窗口警告信息。

6. 因为窗口里是空白的，所以点击 **Yes** 接受。
7. 使用通常会在建模环境中使用的相同的命令和工具来修改视图和渲染动画模式。
8. 点击左下方的分窗口显示 **Driving Torque**（如右图所示）。



9. 点击右上方的分窗口来显示 **Relative Velocity**，作为第一个输出窗口。
10. 点击右下方的分窗口显示车之间的 **Distance**，作为第二个输出窗口。



上图即为最终的结果图像。

此时，已经完成了本教程的主要步骤。附加的步骤包括调整控制增益进一步提高性能或增加控制器的复杂性，使其适应基于汽车之间的实际距离或时间的影响。在下一节中，将完成一个自适应控制系统：当前面汽车离开后，汽车将保持期望速度运行。

Chapter

6

增强 CoLink 模型

任务目标

本章修改 **CoLink** 模型，使其具有自适应巡航控制的功能。仿真系统并绘制结果。

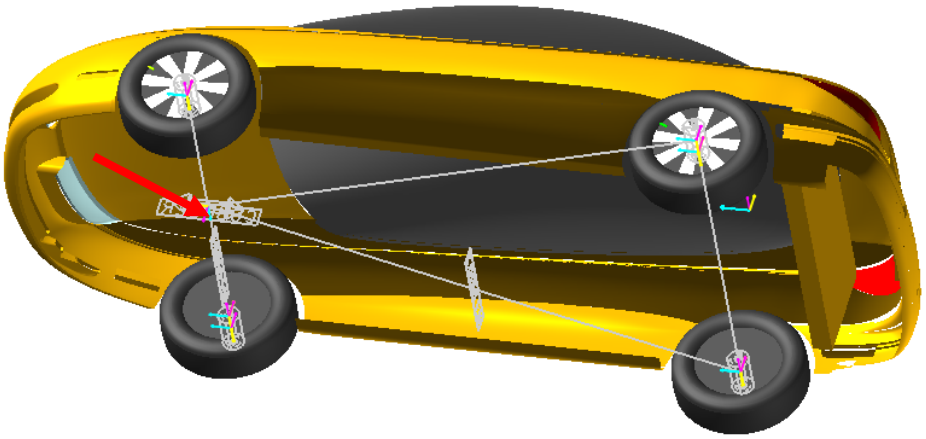
预计完成时间

15 分钟

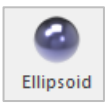
修改 RecurDyn 模型

首先修改 **RecurDyn** 模型，使其实现自适应巡航控制系统。这需要为前车增加一个横向平动副，为模型控制系统增加一个 Plant Output。

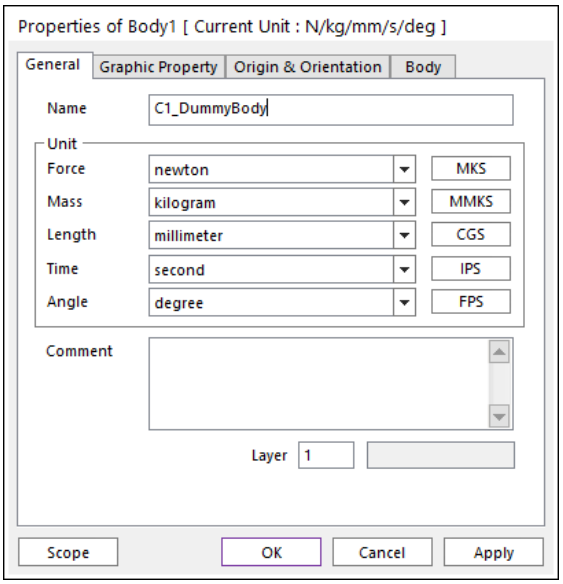
首先要在前车的前轴创建一个哑元。



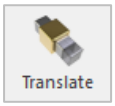
创建哑元：



1. 在 **Professional** 标签下的 **Body** 选项中，点击 **Ellipsoid** 图标。
2. 使用默认创建方式 **Point, Distance**，选择前面车现有平动副和耦合器的坐标，如上图所示。
3. 球的半径输入为 **35**。
4. 右键球体，选择 **Properties**。
5. 在 **General** 页面中，将球体重命名为 **C1_DummyBody**，如右图所示。
6. 点击 **OK**。



创建平动副：



1. 在 **Professional** 标签下的 **Joint** 选项中，点击 **Translate** 图标。
2. 使用 **Body，Body，Point，Direction** 创建方式，依次选择：
   - **Ground**
   - **C1_DummyBody**
   - 球体的中心
   - 输入方向 **0，1，0**（或者点击 **Ground Inertia Marker** 的+**Y** 轴）。
3. 改变新建的运动副的属性。
   - 右键新建的运动副，点击 **Property**。
   - 在 **General** 页面，重命名为 **C1_TraLaneChange**。
   - 在 **Joint** 页面，建立车道变化的方程：

     a. 勾选 **Include Motion** 边上的方框，点击 **Motion**。

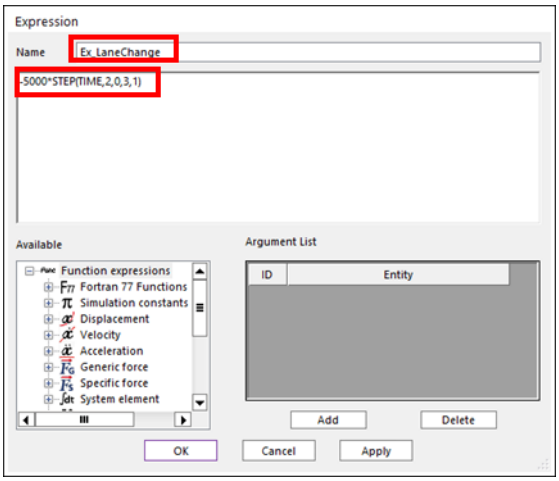
     b. 点击 **EL** 打开表达式窗口，点击 **Create** 创建一个新的表达式。

     c. 将表达式命名为 **Ex_LaneChange**，然后输入下列表达式：

     -5000*STEP(TIME, 2, 0, 3, 1)

     此表达式将使车向右运动 5 米（**Y** 轴负方向），在仿真的第二秒开始，第三秒结束。

     Expression 对话框应与右图相同。

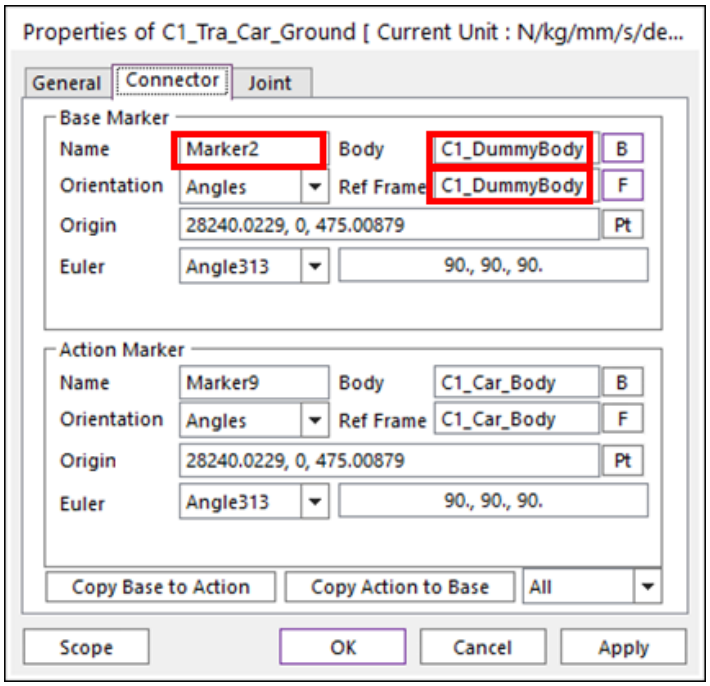


   - 点击 **OK**，保存更改，并退出对话框。

更改原来的平动副，使得车体（**C1_Car_Body**）与球体（**C1_DummyBody**）连接。

4. 在数据库窗口，右键点击 **C1_Tra_Car_Ground**，选择 **Property**。

5. 在 **Connector** 页面，将 **Base Marker Body** 和 **Ref Frame** 改为 **C1_DummyBody**，同时将 **Name** 改为 **Marker2**。因为在数据库窗口中，它是列表中下一个可用的与哑元有联系的 **Marker**（标记），如下图所示。



6. 运行仿真查看是否其正常运行。

5 秒钟的仿真运行，和之前相同，除了运行到一半时，前车会改变车道。

接下来，定义新的 Plant Output，同时扩展 **CoLink** 控制系统，包括新的输出。

定义 Plant Output：

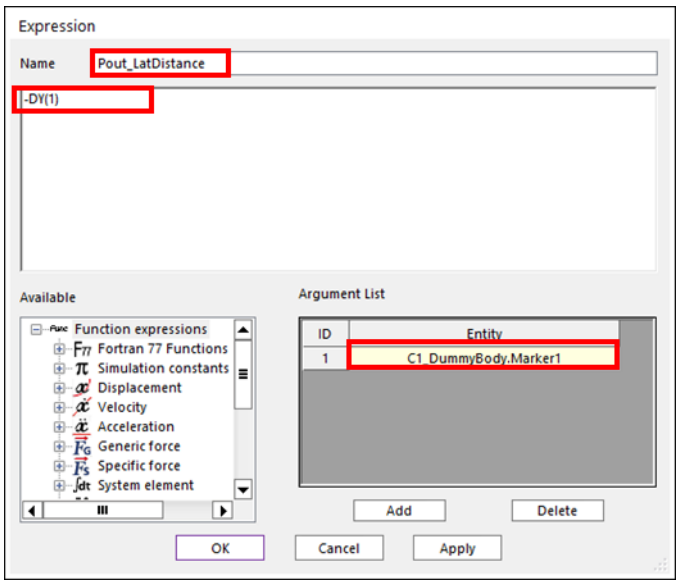
1. 双击现有的 Plant Output，打开 **Plant Output List** 对话框。

2. 点击 **Add**，创建新的输出。

3. 在 **Expression** 对话框中，点击 **Add** 在 **Argumentlist** 中添加一个实体。输入 **C1_DummyBody. Marker1** 或者从数据库窗口中将其拖曳进来

4. 将 **Name** 改为 **Pout_LatDistance**，输入下方的表达式，其中 1 指 **Argument list** 中的条目。

   －DY(1)



5. 点击 **OK**。

6. 在 **Plant Output List** 对话框中，再次点击 **Add**，创建另一个 Plant Output。

7. 将表达式命名为 **Pout_AbsVel**，在 **Argument List** 中输入 **Car_Body. Marker5**，同时输入表达式 **VX(1)**。

   车身上的 **Marker5** 与平动副 **Tra_Car_Ground** 相关联。

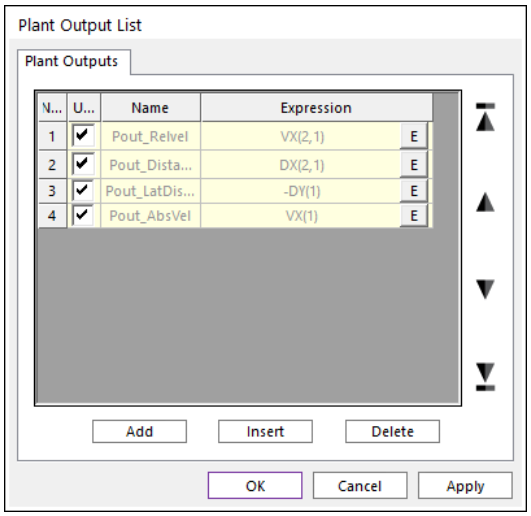
小贴士：可能需要输入一个不同的标记号码，如果采取了额外的措施(例如：创建运动副错误，然后删除并重新创建)，编号会和这里不同。

通过放大数据库中的 **Tra_Car_Ground** 模型，寻找与车身联系的编号的方法，确定正确的编号。

---

8. 点击 **OK**，确定更改，并退出对话框。

Plant Output List 应如右图所示，再次点击 **OK**。

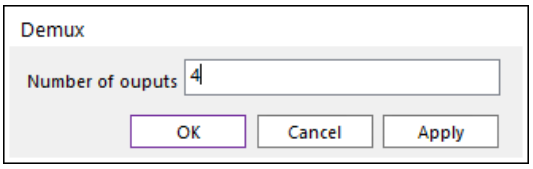


扩展 CoLink 模型

修改 **CoLink** 模型，实现自适应巡航控制。这需要两个运动的汽车之间的横向距离信息和一个额外的反馈控制系统的开关，以保证当前车改道时，汽车能变到理想速度。

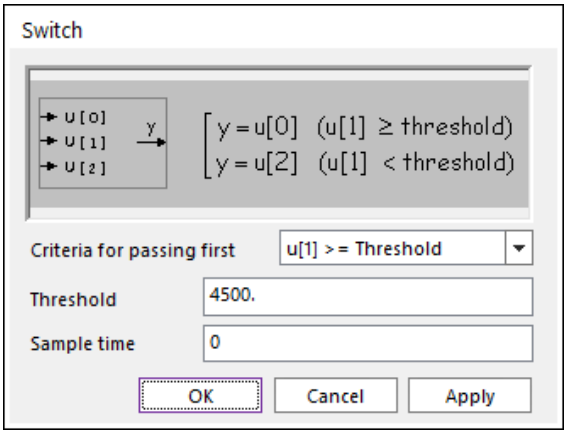
修改 **CoLink** 模型：

1. 打开以前创建的 **CoLink** 模型窗口。
2. 在 **File** 菜单中，点击 **Save As**，将模型以一个新的名字保存。
3. 双击 **Demux** 块，将 **number of outputs** 改为 **4**。



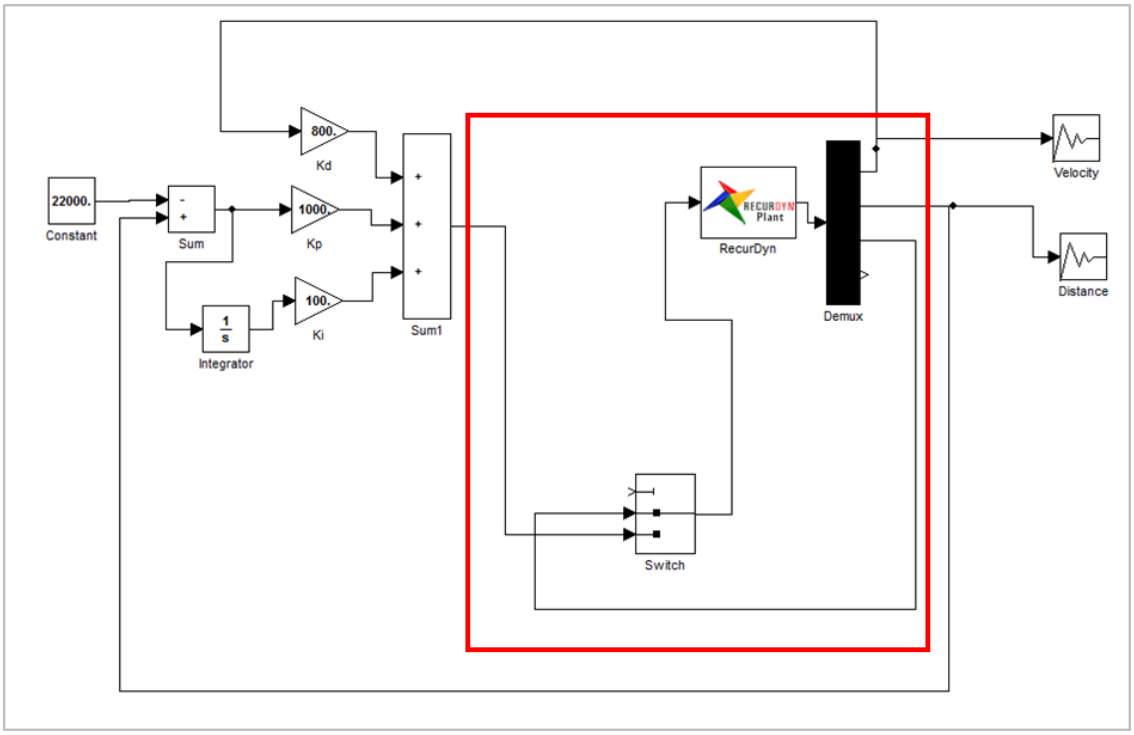
4. 在 **Math** 标签下，点击 **Nonlinear** 中的 **Switch** 在图中创建块。
5. 双击 **Switch** 块，将 **Threshold** 值改为 **4500**。

   Threshold 设为 4500 意味着当第二个输入 **u[1]**大于等于 4. 5 米时，将第一个输入 **u[0]**传递给输出给 **y**

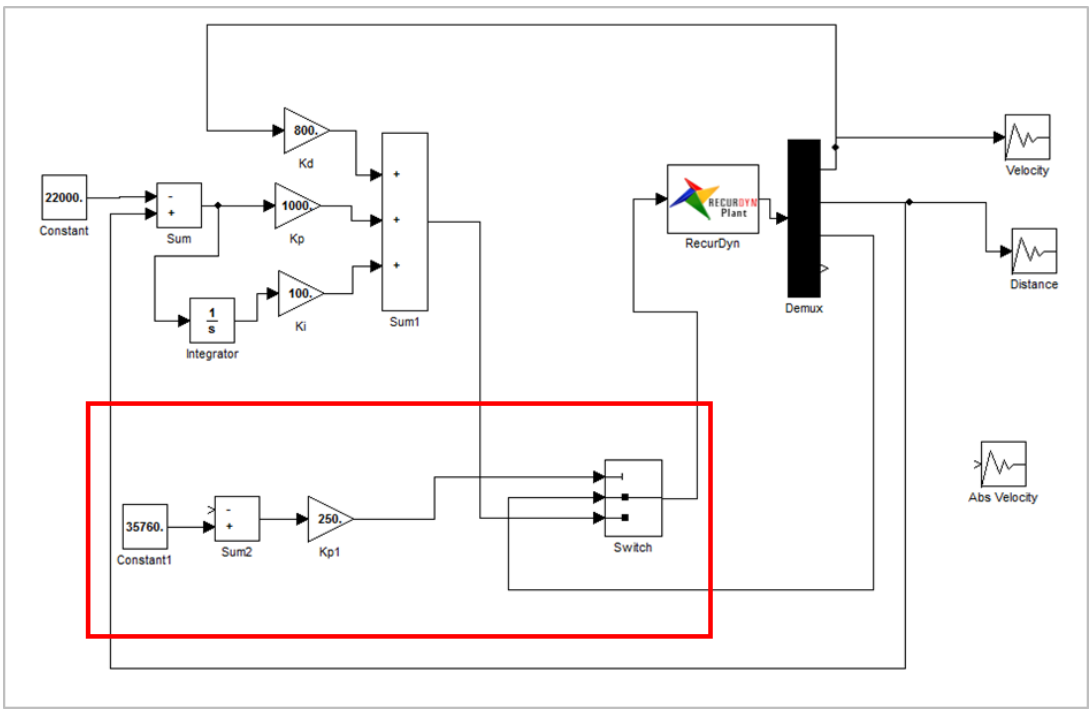


   - **u[1]**为仿真时的两车的横向距离。
   - **u[2]**是两车在同一条车道时的原始参数控制信号。
   - **u[0]**是新控制系统的控制信号，设置期望的驱动速度。

6. 使用下图作为指导，执行以下操作：
   - 删除 **Sum1** 块和 **RecurDyn** 块的连接，将 **Sum1** 与 **Switch** 块的底部连接。
   - 将 **Switch** 块的输出与 **RecurDyn** 块连接，**Demux** 块的第三输出与 **Switch** 块的第二输入连接。



7. 下面，创建新的 **constant** 和 **sum** 块，如下图所示。



点击 **Sum** 块，按 **Ctrl+C** 复制该块，然后按 **Ctrl-V** 粘贴该块，并将其拖到 **Switch** 块的左边。

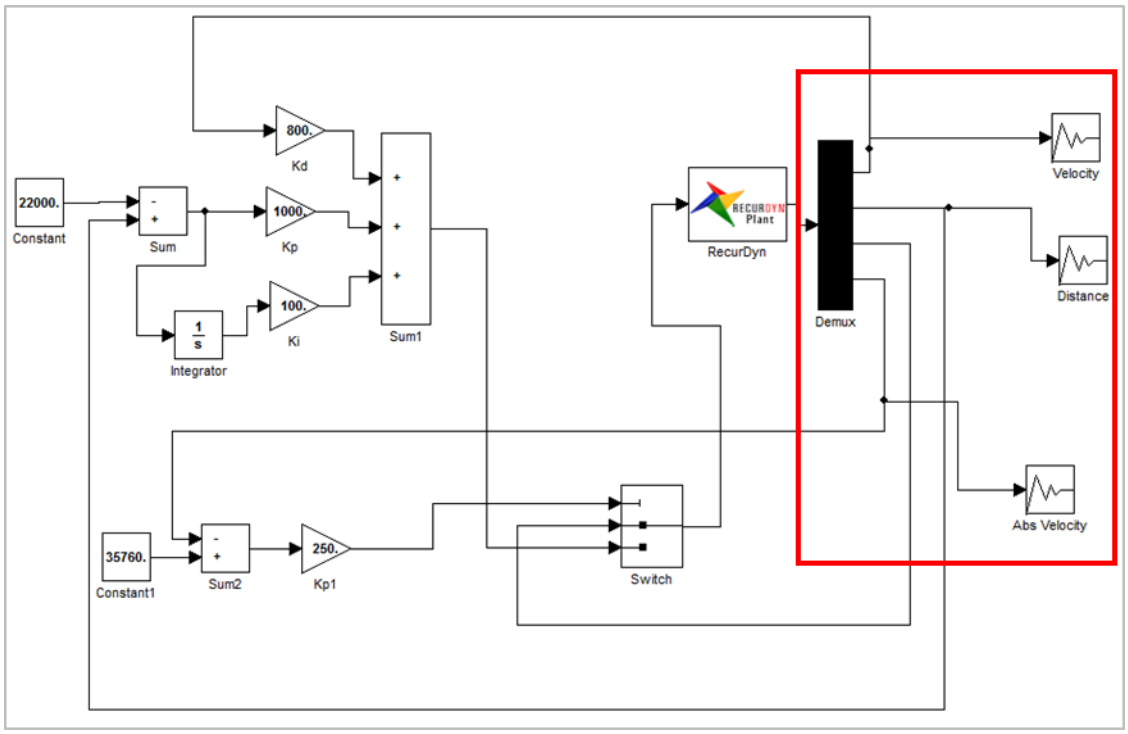
a. 将 **Sum2** 块改为|+，如上图所示。

b. 将 **Demux** 块的第四个输出口与 **Sum2** 块的负端连接。

c. 相似地，复制 **Constant** 和 **Kp** 块，并将它们放在 **Sum2** 块的左边和右边，然后做如下操作。

- 将常数值改为 **35760**（**mm/s**，或者 80**mph**）。
- 将 **Constant2** 与 **Sum2** 块的正端连接。
- 将 **Sum2** 块的输出端与 **Kp1** 的输入端连接。
- 将 **Kp1** 的 **gain** 改为 **250**，并将其输出端与 **Switch** 块的第一个输入端连接。

8. 为 **Demux** 的第四个输出信号添加个 **scope**，命名为 **AbsVelocity**，因为其代表了蓝色车的绝对速度，如下图所示。



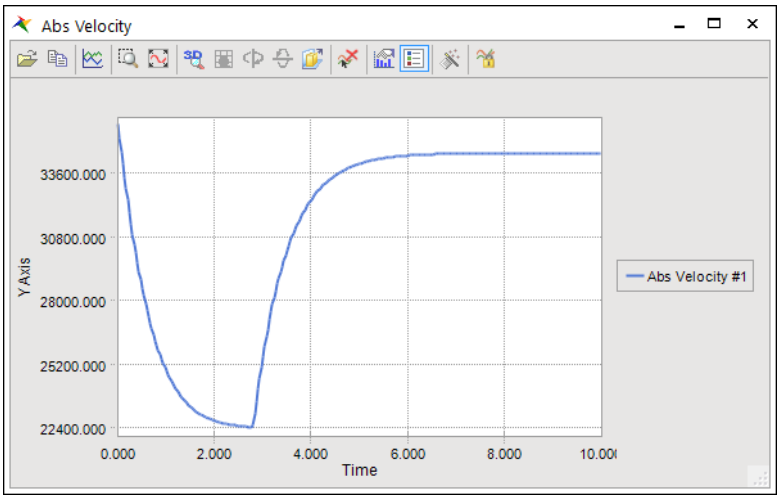
9. 运行仿真 10 秒，确定仿真是否运行正常。

小贴士：改变仿真时间的方法。

在动画控制旁的工具栏中输入 **10**。

点击 **Simulation→Parameter**，将 **End Time** 改为 **10**。

第二辆车的速度应如下图所示。

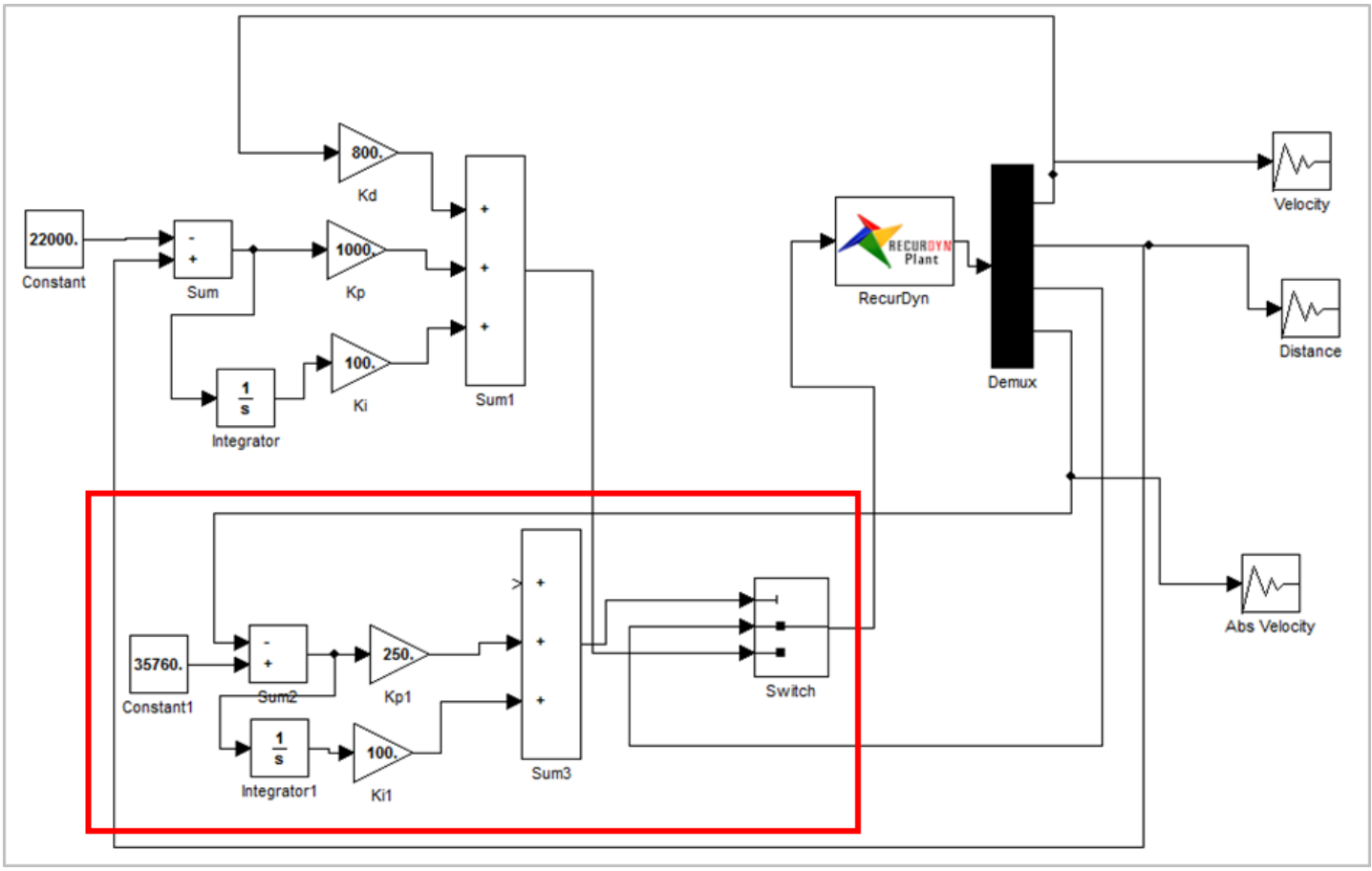


注意，前进的速度并不曾达到开始仿真时的速度。为了纠正这一点，还需要在模拟中添加积分控制器和微分控制器以及速度限制器以便在第 3s 中的加速度。

添加积分和微分控制，如下图所示：

1. 删除 **Kp1** 块与 **Switch** 块的连接。
2. 复制 **Sum1** 块并将其粘贴在 **Kp1** 的右边。
3. 更改 **Sum3** 块的标志为+++，如下图所示。
4. 将 **Kp1** 块与 **Sum3** 块的第二个输入连接。
5. 将 **Sum3** 块的输出口与 **Switch** 块的第一个输入连接。
6. 复制 **Integrator** 和 **Ki** 块，并放在如图所示的位置。将其与之前类似的连接方式，进行连接。

   完成后，模型应如下图所示。



7. 在 **ContinuousandDiscrete** 标签下，点击 **Continuous** 中的 **Derivative** 块，创建块，并放在 **Sum2** 块的上方。

8. 复制 **Kd** 块，并将其放在 **Kp1** 块的上方。

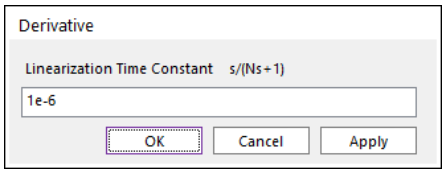
9. 连接如下

   - 将 **Derivative** 块连接在 **Sum2** 和 **Kp1** 块的连线上。
   - 将 **Derivative** 块连接 **Kd1** 块。
   - **Kd1** 块与 **Sum3** 块的第一个输入连接。

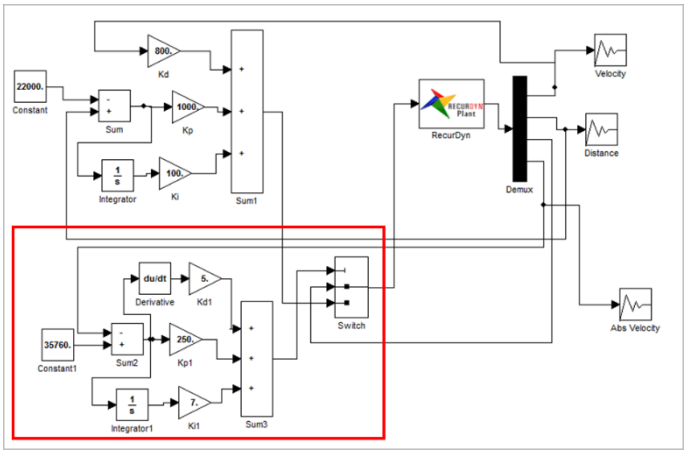
10. 分别将 **Kd1** 和 **Ki1** 的 **gains** 改为 **5** 和 **7**。

    模型应如下图所示。

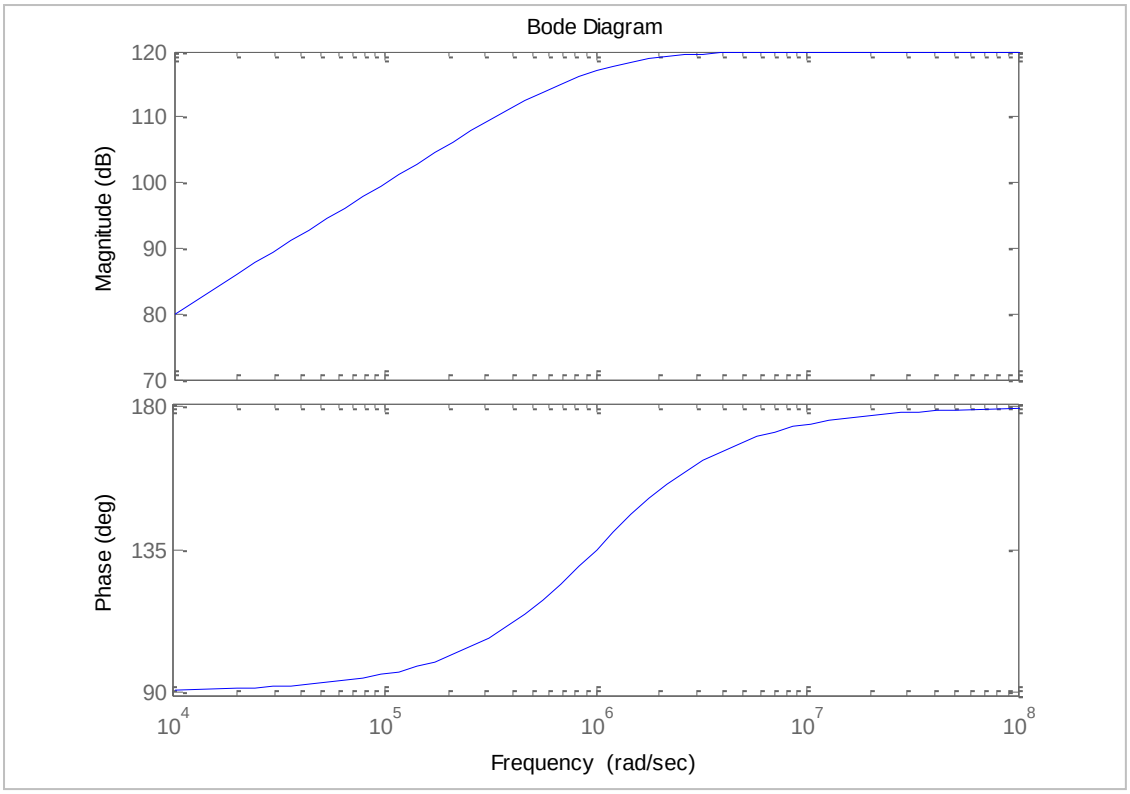
11. 双击 **Derivative** 块，将 **Linearization Time Constant** 改为 **1e−6**。



如右图所示的 **Derivative** 对话框，这个微分器的传递函数现在是 **s**/(1**e**−6**s** + 1)，一个好的微分器的传递函数应为 **s** 或者 **s**/1。虽然这在物理上不可实现，因为在高频率, 增益是无限的。相反, 我

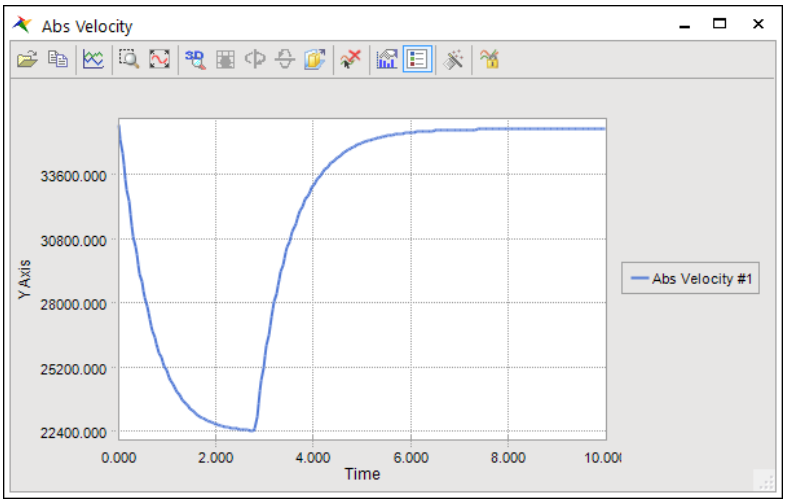


们选择线性化时间常数, 微分器的频率小于 1e-6 **rad/s**，**Bode diagram** 如下图所示：



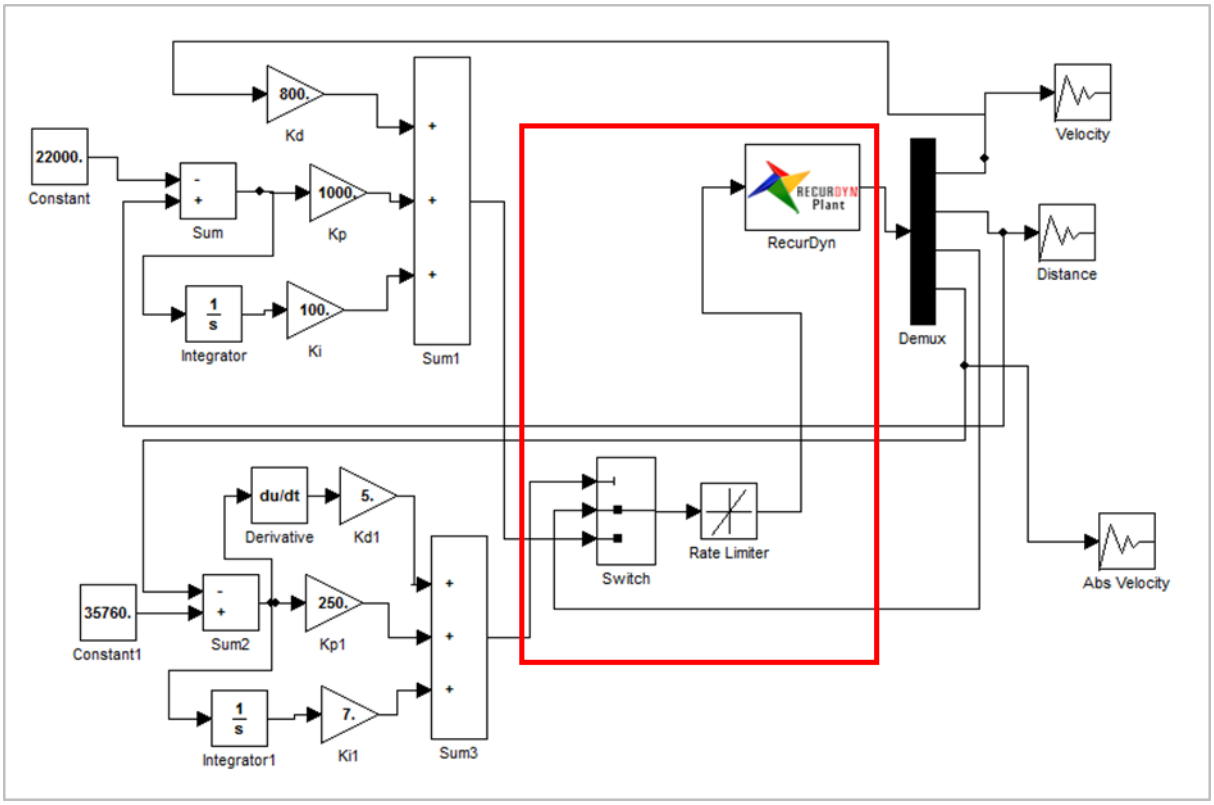
12. 再次仿真模型，确定其是否运行正常。汽车速度现在能达到理想值，其速度图像如下图所示：

接下来，要给控制系统加一个限制器以保证加速度的数值不会太高。特别地，当前车移开后，后车由稳定速度突然加快。这就相当于汽车在急刹车，乘客会非常难受。但很快的减速不是问题，因为当前方有车的话，汽车具有突然减速的能力。

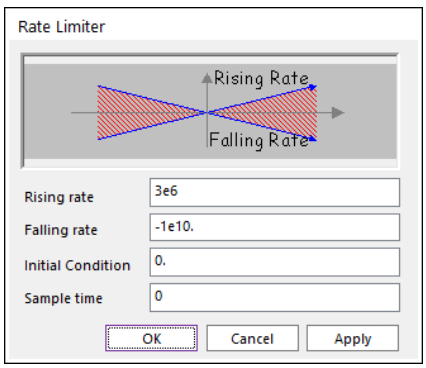


添加限制器，如下图所示：

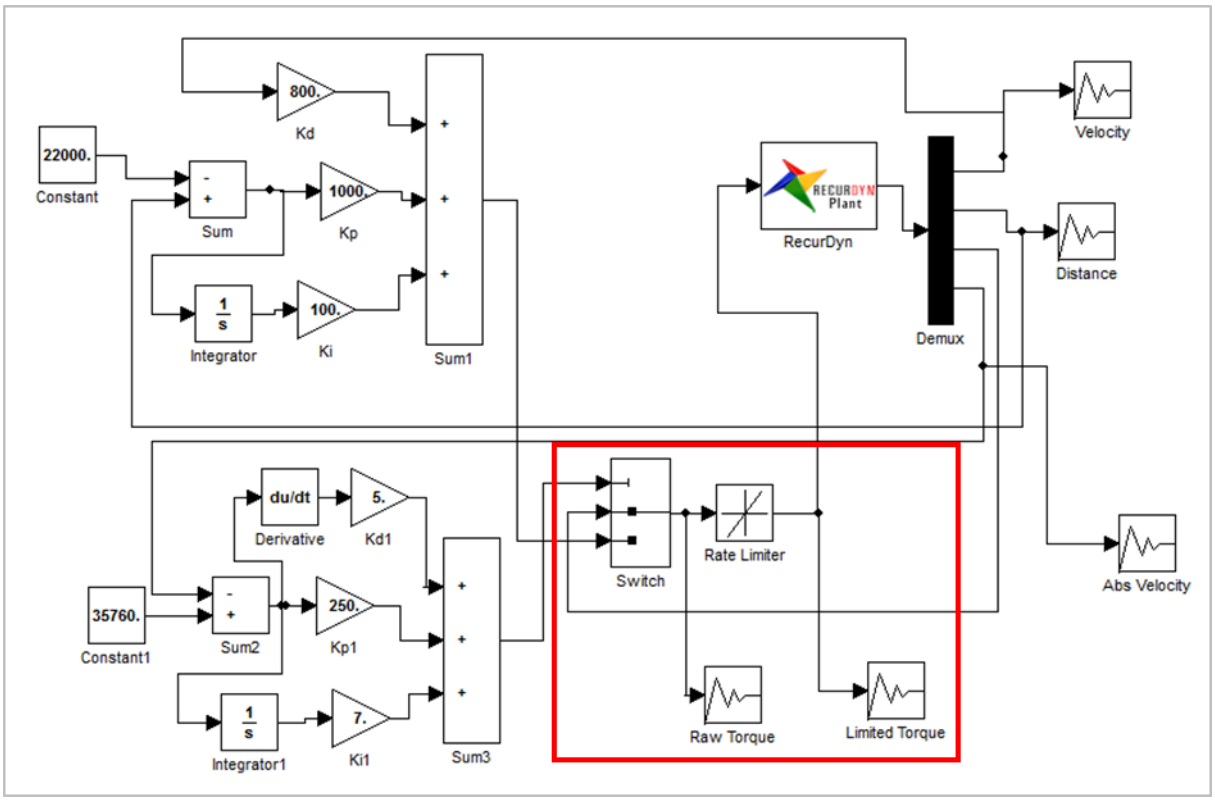
1. 删除 **Switch** 块与 **RecurDyn** 块的连接。
2. 在 **Math** 标签下，点击 **Nonlinear** 中的 **Rate Limiter** 块创建该方快，并放置在 **Switch** 的右边。
3. 将 **Switch** 与 **Rate Limiter** 连接，同时将 **Rate Limiter** 与 **RecurDyn** 块连接，如下图所示。



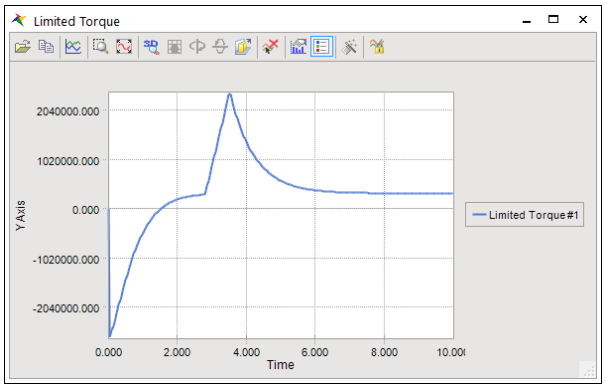
4. 双击 **Rate Limiter** 将 **Rising rate** 改为 **3e6**，将 **Falling rate** 改为**-1e10**。保持初始条件不变。

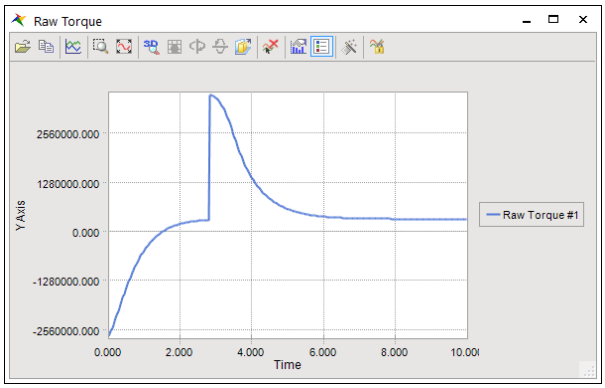


5. 为了观察限制器的效果，在模型中添加两个 **scopes**，将它们连接在限制器的左右，分别将 **Scope** 和 **Scope1** 改为 **RawTorque** 和 **LimitedTorque**。



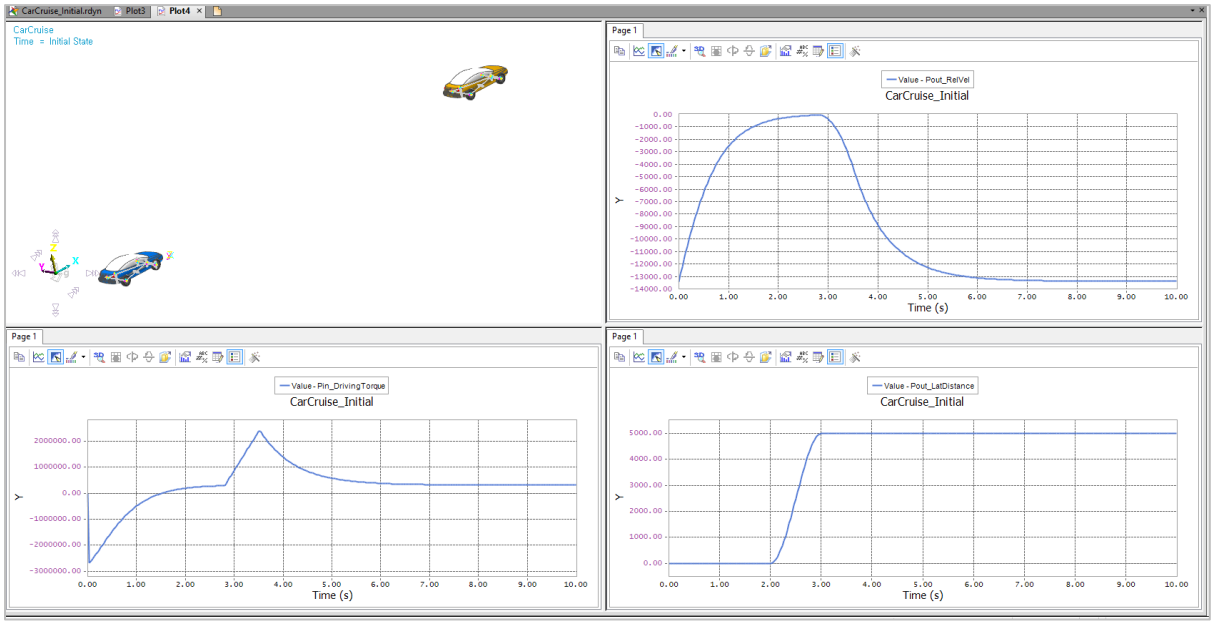
6. 再次仿真模型，双击这两个新的 **scopes**，查看其结果，如下图所示。





7. 打开 **RecurDyn**，按 **Animation Control** 中的 **Play** 键观察仿真动画。

若对仿真结果满意，打开 **plotting environment**，进行与第 5 章最后一样的操作，不过不是绘出两车的相对距离，而是其横向距离，整个图像应如下图所示。



如果有兴趣，可以进一步完善这个模型，尝试以下的其它步骤：

- 调整控制增益来提高仿真性能。
- 将 **Kd** 和 **Ki** 设置为 0，从头开始优化控制器。

感谢参与本教程学习！