ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»

Институт компьютерных наук и технологий

Высшая школа программной инженерии

ПОЛИТЕХ
Санкт-Петербургский политехнический университет Петра Великого

ЛАБОРАТОРНАЯ РАБОТА №1

по дисциплине «Проектирование интеллектуальных систем управления»

Студент гр. 3530202/90202 А. М. Потапова

Руководитель Ю. Н. Кожубаев

Санкт-Петербург
2022 г

Ход работы

В качестве примера мною была выбрана имитация прыгающего мяча.

Модели прыгающего мяча представляют собой гибридные динамические системы с феноменом Зенона. Поведение Зенона неформально характеризуется бесконечным числом событий, происходящих за конечный интервал времени для определенных гибридных систем. По мере того, как мяч теряет энергию, большое количество столкновений с землей начинает происходить через последовательно меньшие промежутки времени.

Модель прыгающего мяча является примером гибридной динамической системы. Гибридная динамическая система — это система, которая включает в себя как непрерывную динамику, так и дискретные переходы, в которых динамика системы может меняться, а значения состояния могут скачком. Непрерывная динамика прыгающего мяча определяется следующими уравнениями:

$$\frac{dv}{dt} = -g, \qquad \frac{dx}{dt} = v,$$

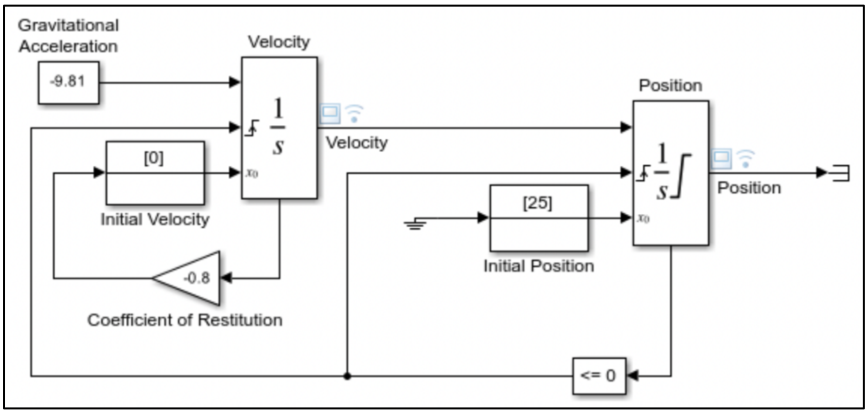
где g — ускорение свободного падения, $x(t)$ — положение мяча, а $v(t)$ — скорость. Следовательно, система имеет два непрерывных состояния: положение x и скорость v.

Аспект гибридной системы модели происходит от моделирования столкновения мяча с землей. Если предположить частично упругое столкновение с землей, то скорость до столкновения — v^-, и скорость после столкновения — v^+, могут быть связаны коэффициентом восстановления шара — k, следующим образом:

$$v^+ = -k \cdot v^-, \qquad x = 0,$$

Таким образом, прыгающий мяч демонстрирует прыжок в непрерывном состоянии (скорость) при условии перехода $x = 0$.

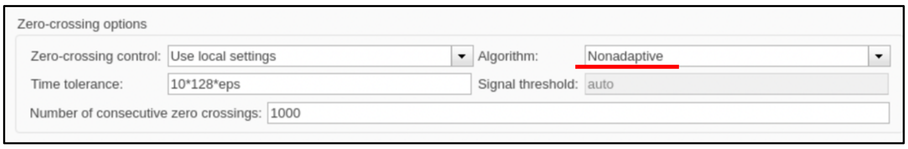
Реализация модели используя два блока итераторов



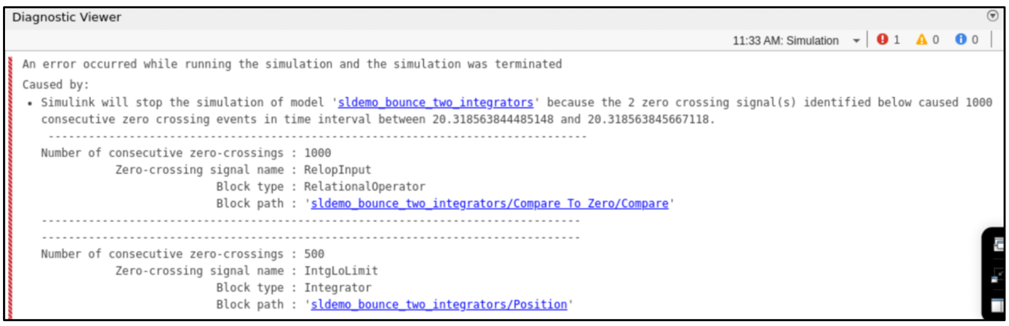
Чтобы наблюдать за поведением системы Zeno, переходим на панель Solver диалогового окна Configuration Parameters. В Simulation time устанавливаем Stop time на 25.

- Во времени моделирования установите время остановки на 25.

- Разверните детали Sovler. В параметрах Zero-crossing установливаем для параметра Algorithm значение Nonadaptive.

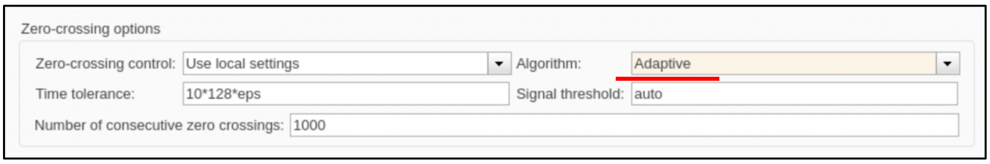


Результаты моделирования:

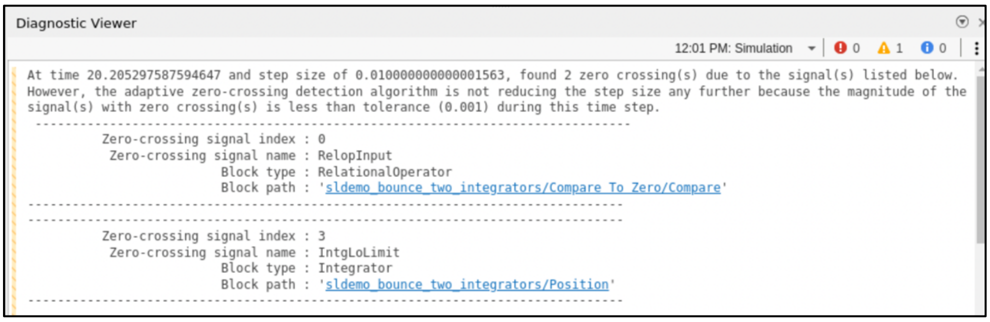


Поскольку мяч чаще ударяется о землю и теряет энергию, симуляция превышает установленный по умолчанию лимит числа последовательных пересечений нуля, равный 1000.

В Solver> Zero-crossing options устанавливаем алгоритм на адаптивный. Этот алгоритм вводит сложную обработку болтливого поведения.



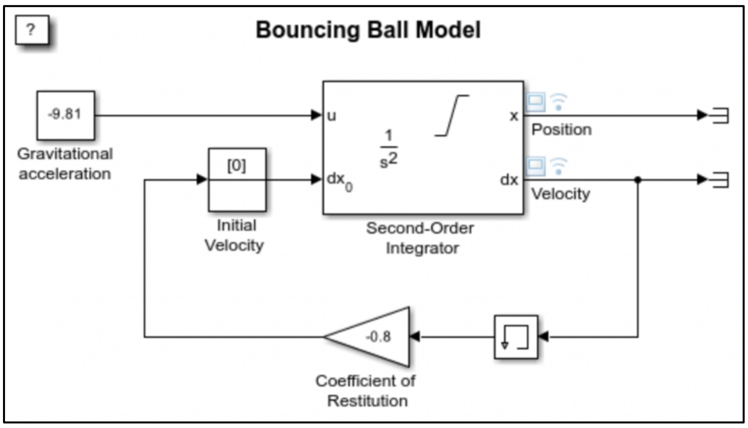
Результаты моделирования:




```
At time 20.205297587594647 and step size of 0.010000000000001563, found 2 zero crossing(s) due to the signal(s) listed below.
However, the adaptive zero-crossing detection algorithm is not reducing the step size any further because the magnitude of the
signal(s) with zero crossing(s) is less than tolerance (0.001) during this time step.
---------------------------------------------------------------------------------
        Zero-crossing signal index : 0
         Zero-crossing signal name : RelopInput
                        Block type : RelationalOperator
                        Block path : 'sldemo_bounce_two_integrators/Compare To Zero/Compare'
---------------------------------------------------------------------------------
---------------------------------------------------------------------------------
        Zero-crossing signal index : 3
         Zero-crossing signal name : IntgLoLimit
                        Block type : Integrator
                        Block path : 'sldemo_bounce_two_integrators/Position'
---------------------------------------------------------------------------------
```

Теперь мы можем смоделировать систему за пределами 20 секунд. Обращаем внимание на дребезг состояний между 21 и 25 секундами и предупреждение от Simulink о сильном дребезге в модели около 20 секунд.

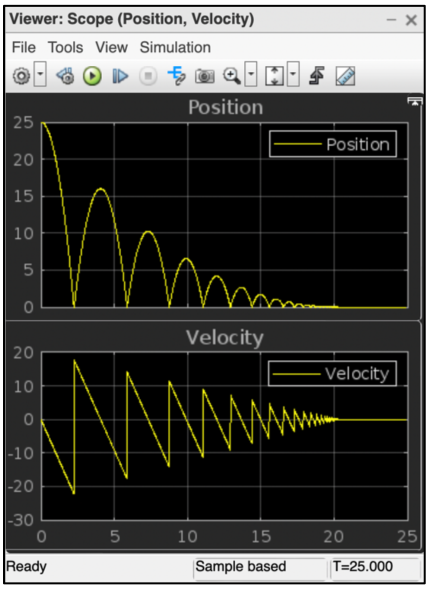
Реализация модели используя блок интегратора второго порядка, чтобы смоделировать прыгающий мяч



- В Simulation time устанавливаем Stop time на 25.

- В параметрах пересечения нуля устанавливаем для алгоритма значение Nonadaptive.

Результаты моделирования:



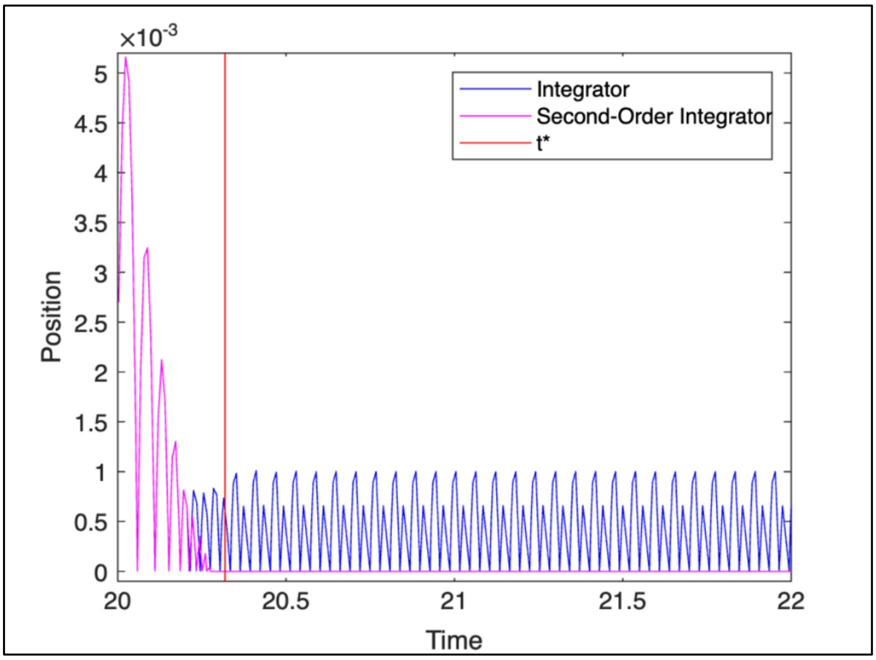
Симуляция не вызывает проблем. Мы можем смоделировать модель, не испытывая чрезмерной вибрации через 20 секунд и не устанавливая для алгоритма значение «Адаптивный».

Сравнение подходов к моделированию прыгающего мяча

Мы можем аналитически рассчитать точное время t^*, когда мяч опустится на землю с нулевой скоростью, просуммировав время, необходимое для каждого отскока. Это время представляет собой сумму бесконечного геометрического ряда, заданного формулой:

$$t^* = \frac{1}{g}\left(v_0 + v_1\left(\frac{1+k}{1-k}\right)\right), \qquad v_1 = \sqrt{v_0^2 + 2gx_0},$$

где x_0 и v_0 — начальные условия для положения и скорости соответственно. Скорость и положение мяча должны быть тождественно равны нулю при $t < t^*$. На рисунке показаны результаты обоих расчетов вблизи t^*. Вертикальная красная линия на графике — это t^*для заданных параметров модели. При $t < t^*$ и вдали от t^*обе модели дают точные и идентичные результаты. На графике видна только пурпурная линия от второй модели. Однако результаты моделирования по первой модели неточны после t^*. На графике по-прежнему наблюдается чрезмерная вибрация для $t < t^*$. Напротив, модель, использующая блок Second-Order Integrator, устанавливается точно на ноль для $t < t^*$.



Вывод

Модель, которая использует блок Second-Order Integrator, имеет превосходящие числовые характеристики по сравнению с первой моделью, потому что второе дифференциальное уравнение $\frac{dx}{dt} = v$ является внутренним для блока Second-Order Integrator. Алгоритмы блоков могут использовать эту взаимосвязь между двумя состояниями и использовать эвристику для подавления дребезга при определенных условиях. Эти эвристики становятся активными, когда два состояния перестают быть взаимно согласованными из-за ошибок интегрирования и вибраций. Таким образом, мы можем использовать физические знания системы, чтобы предотвратить застревание симуляций в состоянии Zeno для определенных классов моделей Zeno.