Задача 13

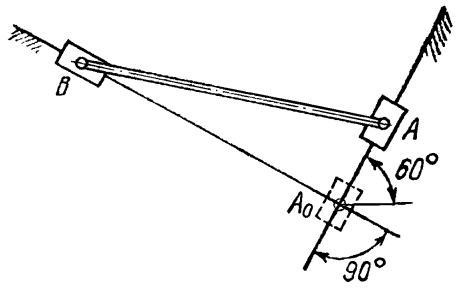
(Задача 9.48. Сборник задач по теоретической механике/ Под ред. К.С. Колесникова. М.: Наука, Главная редакция физико-математической литературы, 1983. – 320 с.)

Ползуны *A* и *B* одинаковой массы *m*, шарнирно соединенные однородным стержнем *AB* длины *l*, имеющим также массу *m*, могут скользить без трения по взаимно перпендикулярным направляющим, расположенным в вертикальной плоскости. В положении A_0 ползуну *A* сообщается начальная скорость v_0.

Определить, при каком значении начальной скорости стержень достигнет горизонтального положения.

Для решения задачи использовать следующие значения параметров:

$m = 1$ *кг*, $l = 0.4$ *м*, $v_0 = 1$ *м/с*.

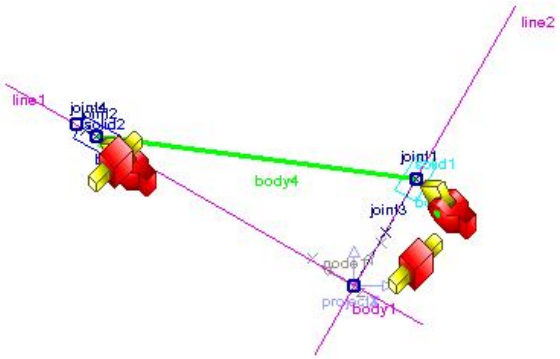


Точный теоретический ответ

Начальная скорость ползуна определяется по формуле

$$v_0 = \frac{3}{4} \cdot \sqrt{2 \cdot g \cdot l \cdot \left(\sqrt{3} - 1\right)}.$$

Решение задачи в EULER



Система состоит из четырех звеньев.

- Инерциальное звено (body1). В проекте звено отображается линиями line1 и line2.
- Два ползуна (body2, body3). В проекте звенья отображаются параллелепипедами solid1 и solid2. Каждый из них имеет массу m.
- Стержень (body4). Звено изображается цилиндром solid3 и имеет массу m.

Ползуны и стержень связаны шарнирами joint1 и joint2 типа «пара вращения». Движение ползунов по направляющим моделируется шарнирами joint3 и joint4 типа «поступательная пара». В проекте задана гравитация с ускорением свободного падения 9.81 [m/ s2]. Начальная скорость первому ползуну сообщается с помощью объекта «условие состояния механизма» condition1. Для определения горизонтального положения стержня используется датчик расстояния между точками y. События event1 останавливает расчет, когда скорость первого ползуна станет равна нулю.

Для решения задачи создана команда «Краевая задача» command2. Эта команда позволяет найти начальную скорость первого ползуна, при которой стержень достигнет

горизонтального положения. Для интегрирования движения системы при решении краевой задачи используется команда «Расчет динамики движения» command1.

Результаты моделирования

Относительное отличие решения задачи в EULER в зависимости от шага численного интегрирования (использовался постоянный шаг интегрирования) представлено в следующей таблице (для значений абсолютной и относительной погрешностей решения краевой задачи 0.00001 [-]).

Шаг интегрирования [s]	Относительное отличие от теоретического решения v0_delta_rel [-]
0.1	0.001 6
0.01	0.000 002 5
0.001	0.000 002 3

Текст проекта в EULER

```
scalar m=1 [ kg ];
scalar l=0.4 [ m ];
scalar v0=1 [ m/ s ];
scalar g=9.81 [ m/ s2 ];
point point1=point( 0 [ m ], 0 [ m ], 0 [ m ] );
node node1=node( point1, 0 [ rad ], 0 [ rad ], 60 [ deg ] );
point point2=pointN( node1, 0 [ m ], 0.5 [ m ], 0 [ m ] );
point point3=pointN( node1, 0 [ m ], -0.1 [ m ], 0 [ m ] );
point point4=pointN( node1, 0.4 [ m ], 0 [ m ], 0 [ m ] );
point point5=pointN( node1, -0.1 [ m ], 0 [ m ], 0 [ m ] );
line line1=polyLine( list( point2, point3 ) );
line line2=polyLine( list( point4, point5 ) );
body body1=body( color = RGB( 229, 0, 229 ) );
set ground = body1;
body body1 < ( line1, line2 );
point point6=pointN( node1, 0 [ m ], l, 0 [ m ] );
solid solid1=box( node1, 0.05 [ m ], 0.03 [ m ], 0.03 [ m ], mass = m );
solid solid2=box( nodePN( point6, node1 ), 0.03 [ m ], 0.05 [ m ], 0.03 [ m ], mass = m );
solid solid3=cylinder( point1, point6, 0.002 [ m ], mass = m );
body body2=body( color = RGB( 0, 255, 255 ) );
body body2 < ( solid1 );
body body3=body( color = RGB( 0, 0, 255 ) );
body body3 < ( solid2 );
body body4=body( color = RGB( 0, 255, 0 ) );
body body4 < ( solid3 );
joint joint1=rotational( body2, body4, point1, projectZ );
joint joint2=rotational( body3, body4, point6, projectZ );
joint joint3=translational( body1, body2, point1, vectorX( node1 ) );
joint joint4=translational( body1, body3, point6, vectorY( node1 ) );
gravity gravity1=parallel( reverse( projectY ), g = g );
condition condition1=transVelocity( body1, vectorX( node1 ), body2, point1, v0 );
sensor y=bodyDisplacement( body2, point1, projectY, body3, point6 );
sensor v=derivative( joint3.s );
event event1=reformsBySensor( list( stop(  ) ), v, 0 [ m/ s ], gauge = list(  ) );
sensor v0_theoretical=3/4*sqrt( 2*g*l*( sqrt( 3 ) -1 ) );
sensor v0_delta_rel=abs( ( v0 - v0_theoretical )/ v0_theoretical );
command command1=constRK4( 1.00000e+000 [ s ], 1.00000e-003 [ s ] );
command command2=boundaryProblem( list( v0 ), list( y ), list( 0.0  [ m ] ), 0.00001, 0.00001, 100,
command1 );

/\////////////////////////////////////////////////////////////////////////////
/\  Единицы измерения;
set units = SI;
```