

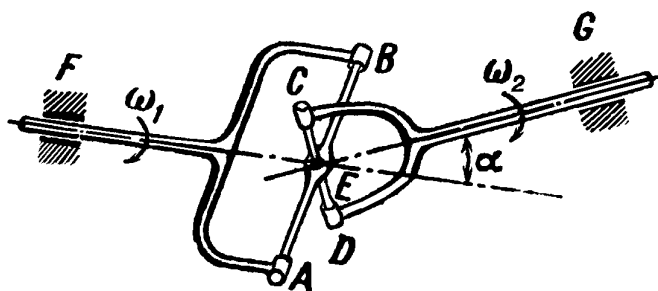
## Задача 12

( Задача 24.41. Задачи по теоретической механике. И.В. Мещерский. СПб.: Издательство «Лань», 2001. – 448 с. )

Крестовина  $ABCD$  универсального шарнира Кардана-Гука ( $AB \perp CD$ ), употребляемого при передаче вращения между пересекающимися осями, вращается вокруг неподвижной точки  $E$ . Найти отношение  $\omega_1/\omega_2$  для валов, связанных крестовиной в двух случаях:

- 1) когда плоскость вилки  $ABF$  горизонтальна, а плоскость вилки  $CDG$  вертикальна;
- 2) когда плоскость вилки  $ABF$  вертикальна, а плоскость вилки  $CDG$  ей перпендикулярна.

Угол между осями валов постоянный:  $\alpha = 60^\circ$ .

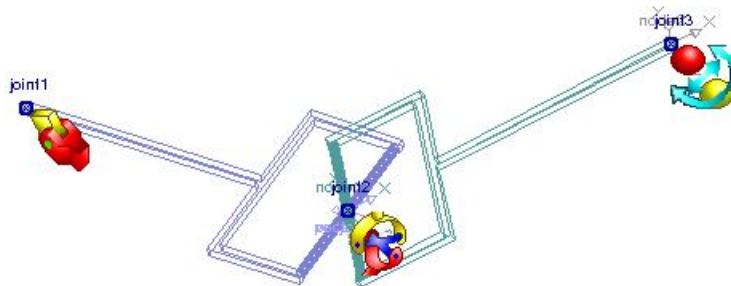


### Точный теоретический ответ

Отношение угловых скоростей валов определяется по формулам:

- 1)  $\omega_1/\omega_2 = 1 / \cos \alpha$ ;
- 2)  $\omega_1/\omega_2 = \cos \alpha$ .

### Решение задачи в EULER



Система состоит из трех звеньев:

- Вал 1 (body1). Звено отображается телами solid1 и solid2.
- Вал 2 (body2). Звено отображается телами solid3 и solid4.
- Инерциальное звено (body3). Звено в проекте не отображается.

Вращение первого вала моделируется шарниром joint1 типа «пара вращения».

Вращение второго вала моделируется шарниром joint3 типа «пользовательский шарнир», который имеет все три вращательные степени свободы одну поступательную, вдоль оси вала. Поступательная и вращательные степени свободы второго вала позволяют избежать заклинивания механизма. Передача вращения от первого вала ко второму моделируется шарниром joint2 типа «кардановый шарнир».

Начальная угловая скорость первому валу сообщается с помощью объекта «условие состояния механизма» condition1. Для определения угловых скоростей валов используются датчики w1 и w2. Отношение угловых скоростей валов вычисляется с помощью датчика w1\_w2.

В проекте создан объект «условие состояния механизма» condition2. Он изменяет положение механизма, которое соответствует второму условию задачи.

Для решения задачи в первом случае достаточно перейти в режим исследования проекта. При этом condition2 должен находиться в режиме «включен».

Для решения задачи во втором случае необходимо переключить состояние объекта condition2 в режим «выключен» и перейти в режим исследования проекта.

### **Результаты моделирования**

При моделировании в EULER для принятых значений параметров получено.

В первом случае относительное отличие от теоретического решения

$$wa\_delta\_rel = 0.000\ 000\ 000\ 17 [-].$$

Во втором случае относительное отличие от теоретического решения

$$wb\_delta\_rel = 0.000\ 000\ 000\ 85 [-].$$

### **Текст проекта в EULER**

```
scalar a=60 [ deg];
scalar w0=10 [deg/ s ];
point point1=point( 0.01 [ m ], 0 [ m ], 0.2 [ m ] );
point point2=point( 0.01 [ m ], 0 [ m ], -0.2 [ m ] );
point point3=point( 0.01 [ m ], 0 [ m ], 0.22 [ m ] );
point point4=point( 0.01 [ m ], 0 [ m ], -0.22 [ m ] );
point point5=point( -0.22 [ m ], 0 [ m ], 0.22 [ m ] );
point point6=point( -0.22 [ m ], 0 [ m ], -0.22 [ m ] );
point point7=point( -0.22 [ m ], 0 [ m ], 0.01 [ m ] );
point point8=point( -0.22 [ m ], 0 [ m ], -0.01 [ m ] );
point point9=point( -0.8 [ m ], 0 [ m ], 0.01 [ m ] );
point point10=point( -0.8 [ m ], 0 [ m ], -0.01 [ m ] );
point point11=point( -0.2 [ m ], 0 [ m ], 0.2 [ m ] );
point point12=point( -0.2 [ m ], 0 [ m ], -0.2 [ m ] );
line line1=polyLine( list( point1, point3, point5, point7, point9, point10, point8, point6, point4, point2,
point12, point11, point1 ) );
point point13=point( 0.01 [ m ], 0.01 [ m ], 0.2 [ m ] );
point point14=point( 0.01 [ m ], -0.01 [ m ], 0.2 [ m ] );
solid solid1=extrude( line1, point13, point14, mass = 1 [ g ] );
point point15=point( 0 [ m ], 0 [ m ], 0.2 [ m ] );
point point16=point( 0 [ m ], 0 [ m ], -0.2 [ m ] );
solid solid2=cylinder( point15, point16, 0.01 [ m ] );
color color1=RGB( 123, 128, 210 );
body body1=body( color = color1 );
body body1 < ( solid1, solid2 );
point point17=point( 0 [ m ], 0 [ m ], 0 [ m ] );
node node1=node( point17, 0 [ deg ], a, 0 [ deg ] );
point point18=pointN( node1, -0.01 [ m ], 0.2 [ m ], 0 [ m ] );
point point19=pointN( node1, -0.01 [ m ], -0.2 [ m ], 0 [ m ] );
point point20=pointN( node1, -0.01 [ m ], 0.22 [ m ], 0 [ m ] );
point point21=pointN( node1, -0.01 [ m ], -0.22 [ m ], 0 [ m ] );
point point22=pointN( node1, 0.22 [ m ], 0.22 [ m ], 0 [ m ] );
point point23=pointN( node1, 0.22 [ m ], -0.22 [ m ], 0 [ m ] );
point point24=pointN( node1, 0.22 [ m ], 0.01 [ m ], 0 [ m ] );
point point25=pointN( node1, 0.22 [ m ], -0.01 [ m ], 0 [ m ] );
point point26=pointN( node1, 0.8 [ m ], 0.01 [ m ], 0 [ m ] );
point point27=pointN( node1, 0.8 [ m ], -0.01 [ m ], 0 [ m ] );
point point28=pointN( node1, 0.2 [ m ], 0.2 [ m ], 0 [ m ] );
point point29=pointN( node1, 0.2 [ m ], -0.2 [ m ], 0 [ m ] );
line line2=polyLine( list( point18, point20, point22, point24, point26, point26, point27, point25, point23,
point21, point19, point29,
point28, point18 ) );
point point30=pointN( node1, 0 [ m ], 0.2 [ m ], 0.01 [ m ] );
point point31=pointN( node1, 0 [ m ], 0.2 [ m ], -0.01 [ m ] );
solid solid3=extrude( line2, point30, point31, mass = 1[ g ] );
point point32=pointN( node1, 0 [ m ], 0.2 [ m ], 0 [ m ] );
point point33=pointN( node1, 0 [ m ], -0.2 [ m ], 0 [ m ] );
solid solid4=cylinder( point32, point33, 0.01 [ m ] );
color color2=RGB( 78, 156, 156 );
body body2=body( color = color2 );
```

```

body body2 < ( solid3, solid4 );
body body3=body( color = RGB( 229, 229, 229 ) );
set ground = body3;
point point34=point( -0.8 [ m ], 0 [ m ], 0 [ m ] );
joint joint1=rotational( body3, body1, point34, projectX );
point point35=pointN( node1, 0.8 [ m ], 0 [ m ], 0 [ m ] );
vector vector1=vectorX( node1 );
vector vector2=rotate( projectY, projectX, 45 [ deg ] );
joint joint2=cardan( body1, body2, point17, projectX, vector1, projectZ );
node node2=nodePN( point35, node1 );
joint joint3=user( body3, body2, node2, yesX:, noY:, noZ:, yesFiX:, yesFiY:, yesFiZ: );
condition condition1=rotVelocity( body3, projectX, body1, w0 );
condition condition2=sensorValue( joint1.gamma, 90 [ deg ], work = off: );
sensor w1=rotVelocity( body3, projectX, body1 );
sensor w2=rotVelocity( body3, vector1, body2, fixingVector = unlockVector: );
sensor w1_w2=w1/w2;
sensor wa_theoretical=1/cos( a );
sensor wa_delta_rel=abs( (w1_w2-wa_theoretical)/ wa_theoretical );
sensor wb_theoretical=cos( a );
sensor wb_delta_rel=abs( (w1_w2-wb_theoretical)/ wb_theoretical );

```

Λ//

Λ Единицы измерения;

set units = SI;