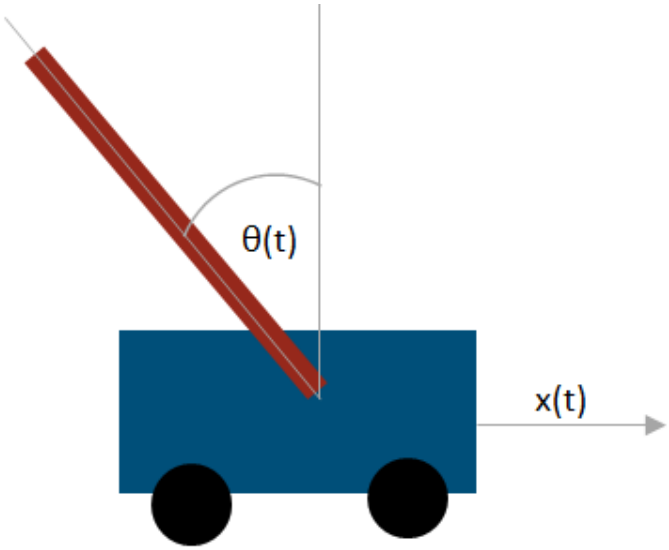
LQR Controller for an Inverted Pendulum on a Cart

Introduction

This worksheet derives the equations that describe the dynamics of an inverted pendulum on a cart, creates a linear quadratic state (LQR) controller that stabilizes the position of the pendulum, and animates the motion of the controlled cart.



Note: This application uses the LQR command from optional Control Design Toolbox to calculate the parameters of an LQR controller. The parameters generated by this command are copied into this application. This allows you to run the application without the toolbox.

```
> restart :
  with( DynamicSystems ) :
  with( LinearAlgebra ) :
```

Model Of Inverted Pendulum on Cart

We will use the following variables and constants in the analysis.

- $x(t)$ is the position of the cart

- $\theta(t)$ is the counter-clockwise angular displacement of the pendulum from the upright position
- $\phi(t)$ is the angular velocity of the pendulum
- u(t) is horizontal force applied to the cart
- L is the half-length of the pendulum
- m is the mass of the pendulum
- M is the mass of the cart
- P is the downward force exerted by the pendulum on the cart
- N is the horizontal force exerted by the pendulum on the cart
- g is the gravitational constant

Let $y(t) = \frac{d}{dt} x(t)$ and $\phi(t) = \frac{d}{dt} \text{theta}(t)$.

> EQ1 := $\dot{x}(t) = y(t)$:
EQ2 := $\dot{\theta}(t) = \phi(t)$:

r is the position of the center of mass of the pendulum.

> r := $[x(t) - L \cdot \sin(\theta(t)), L \cdot \cos(\theta(t))]$:

The acceleration of the center of mass is then:

> acc := $\frac{d^2}{dt^2} r$

$$acc := \left[\ddot{x}(t) - L\ddot{\theta}(t)\cos(\theta(t)) + L\dot{\theta}(t)^2\sin(\theta(t)),\ -L\ddot{\theta}(t)\sin(\theta(t)) - L\dot{\theta}(t)^2\cos(\theta(t))\right] \quad \textbf{(2.1)}$$

Rewrite the acceleration in terms of the cart velocity and angular velocity of pendulum.

> rpp := subs([EQ1, EQ2], acc)

$$rpp := \left[\dot{y}(t) - L\dot{\phi}(t)\cos(\theta(t)) + L\phi(t)^2\sin(\theta(t)),\ -L\dot{\phi}(t)\sin(\theta(t)) - L\phi(t)^2\cos(\theta(t))\right] \quad \textbf{(2.2)}$$

Apply F=ma in the horizontal (x-direction) to the pendulum.

> eq1 := $N = m \cdot rpp_1$

$$eq1 := N = m\left(\dot{y}(t) - L\dot{\phi}(t)\cos(\theta(t)) + L\phi(t)^2\sin(\theta(t))\right)$$

Apply F=ma in the direction perpendicular to the pendulum.

> eq2 := $(P + g)\sin(\theta(t)) - N\cos(\theta(t)) = mL\dot{\phi}(t) - m\dot{y}(t)$

$$eq2 := (P + g)\sin(\theta(t)) - N\cos(\theta(t)) = mL\dot{\phi}(t) - m\dot{y}(t)$$

Apply F=ma in the horizontal direction to the cart.

> eq3 := u(t) − N = M ẏ(t)

$$eq3 := u(t) - N = M\dot{y}(t)$$

Apply M = I alpha to the pendulum, where M is the sum of all moments about the pendulum's center of mass, I is the pendulum's moment of inertia, and alpha is its angular acceleration.

> inertia := $\frac{1}{3} m L^2$:

eq4 := P L sin(θ(t)) − N L cos(θ(t)) = inertia · $\dot{\phi}(t)$

$$eq4 := P L \sin(\theta(t)) - N L \cos(\theta(t)) = \frac{m L^2 \dot{\phi}(t)}{3}$$

> eqns := solve([eq1, eq2, eq3, eq4], [N, P, $\dot{\phi}(t)$, $\dot{y}(t)$])[ ]

$$eqns := \left[N = \frac{1}{-3\cos(\theta(t))\, m + 2M + 2m} \left(2 L M \phi(t)^2 \sin(\theta(t))\, m - 3 M \cos(\theta(t)) \sin(\theta(t))\, g - 3 u(t) \cos(\theta(t))\, m + 2 u(t)\, m \right), P = \frac{1}{\left(-3\cos(\theta(t))\, m + 2M + 2m\right) \sin(\theta(t))} \left(2 L M \cos(\theta(t))\, \phi(t)^2 \sin(\theta(t))\, m - L \phi(t)^2 \sin(\theta(t))\, m^2 - 3 M \cos(\theta(t))^2 \sin(\theta(t))\, g - 3 u(t) \cos(\theta(t))^2 m + \sin(\theta(t))\, g M + 2 u(t) \cos(\theta(t))\, m + \sin(\theta(t))\, g m + u(t)\, m \right), \dot{\phi}(t) = -\frac{3 \left(L \phi(t)^2 \sin(\theta(t))\, m^2 - \sin(\theta(t))\, g M - \sin(\theta(t))\, g m - u(t)\, m \right)}{\left(-3\cos(\theta(t))\, m + 2M + 2m\right) m L}, \dot{y}(t) = -\frac{2 L \phi(t)^2 \sin(\theta(t))\, m - 3 \cos(\theta(t)) \sin(\theta(t))\, g - 2 u(t)}{-3\cos(\theta(t))\, m + 2M + 2m} \right]$$

> EQ3 := $\dot{y}(t)$ = eval($\dot{y}(t)$, eqns)

$$EQ3 := \dot{y}(t) = -\frac{2 L \phi(t)^2 \sin(\theta(t))\, m - 3 \cos(\theta(t)) \sin(\theta(t))\, g - 2 u(t)}{-3\cos(\theta(t))\, m + 2M + 2m} \qquad \textbf{(2.3)}$$

> EQ4 := $\dot{\phi}(t)$ = eval($\dot{\phi}(t)$, eqns)

$$EQ4 := \dot{\phi}(t) = -\frac{3 \left(L \phi(t)^2 \sin(\theta(t))\, m^2 - \sin(\theta(t))\, g M - \sin(\theta(t))\, g m - u(t)\, m \right)}{\left(-3\cos(\theta(t))\, m + 2M + 2m\right) m L}$$

The final nonlinear model is:

```
> sysEqs := Vector( [EQ1, EQ2, EQ3, EQ4] )
```

$$sysEqs := \begin{bmatrix} \dot{x}(t) = y(t) \\ \dot{\theta}(t) = \phi(t) \\ \dot{y}(t) = -\dfrac{2L\phi(t)^2 \sin(\theta(t))\, m - 3\cos(\theta(t))\sin(\theta(t))\, g - 2u(t)}{-3\cos(\theta(t))\, m + 2M + 2m} \\ \dot{\phi}(t) = -\dfrac{3\left(L\phi(t)^2 \sin(\theta(t))\, m^2 - \sin(\theta(t))\, gM - \sin(\theta(t))\, gm - u(t)\, m\right)}{\left(-3\cos(\theta(t))\, m + 2M + 2m\right) mL} \end{bmatrix}$$

Linearization and State Space Model

Assign values to the parameters.

```
> param_values := g = 9.8, M = 10., m = 2., L = 3.0 :
  assign( param_values )
```

The linearization point:

```
> lin_point := [x(t) = 0, y(t) = 0, theta(t) = 0, u(t) = 0, phi(t) = 0] :
```

Linearize the system and create a state-space model.

```
> sysLin := Linearize( convert( sysEqs, list ), [u(t)], [x(t), theta(t)], lin_point )
```

$$sysLin := \begin{bmatrix} \textbf{State Space} \\ \text{continuous} \\ \text{2 output(s); 1 input(s); 4 state(s)} \\ \text{inputvariable} = [u_1(t)] \\ \text{outputvariable} = [y_1(t), y_2(t)] \\ \text{statevariable} = [x_1(t), x_2(t), x_3(t), x_4(t)] \end{bmatrix}, \left[x_1(t) = \phi(t), x_2(t) = \theta(t), x_3(t) = x(t), x_4(t) = y(t)\right], \left[u_1(t) = u(t)\right], \left[y_1(t) = \theta(t), y_2(t) = x(t)\right] \quad \textbf{(3.1)}$$

Note that the states are in the order given in z.

```
> z := ⟨φ(t), θ(t), x(t), y(t)⟩ :
```

Hence, the A and B matrices are:

```
> A := sysLin[1]:-a
```

(3.2)

$$A := \begin{bmatrix} 0.0 & 3.266666667 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \\ 0.0 & 1.633333333 & 0.0 & 0.0 \end{bmatrix} \quad \textbf{(3.2)}$$

> B := sysLin[1]:-b

$$B := \begin{bmatrix} 0.05555555556 \\ 0.0 \\ 0.0 \\ 0.1111111111 \end{bmatrix} \quad \textbf{(3.3)}$$

The transfer function of the system is:

> sysTF := TransferFunction(sysLin[1]);
tf := sysTF:-tf

$$sysTF := \begin{bmatrix} \textbf{Transfer Function} \\ \text{continuous} \\ \text{2 output(s); 1 input(s)} \\ \text{inputvariable} = [u_1(s)] \\ \text{outputvariable} = [y_1(s), y_2(s)] \end{bmatrix}$$

$$tf := \begin{bmatrix} \dfrac{0.0555555555599999973}{s^2 - 3.26666666699999997} \\ \dfrac{0.1111111111 s^2 - 0.2722222222}{s^4 - 3.266666667 s^2} \end{bmatrix} \quad \textbf{(3.4)}$$

Controllability and Observability Matrices

> CC := ControllabilityMatrix(sysLin[1])

$$CC := \quad \textbf{(4.1)}$$

$$\begin{bmatrix} 0.05555555556 & 0.0 & 0.1814814815145185 & 0.0 \\ 0.0 & 0.05555555556 & 0.0 & 0.1814814815145185 \\ 0.0 & 0.1111111111 & 0.0 & 0.09074074072948148 \\ 0.1111111111 & 0.0 & 0.09074074072948148 & 0.0 \end{bmatrix}$$

> OO := ObservabilityMatrix(sysLin[1])

$$OO := \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1. & 0. & 0. & 0. \\ 0. & 0. & 0. & 1. \\ 0. & 3.26666666699999997 & 0. & 0. \\ 0. & 1.63333333299999994 & 0. & 0. \\ 3.26666666699999997 & 0. & 0. & 0. \\ 1.63333333299999994 & 0. & 0. & 0. \end{bmatrix} \quad \textbf{(4.2)}$$

The controllability and observability matrices have full rank; hence, the system is controllable and observable.

```
> Rank(CC);
  Rank(OO)
```

$$4$$
$$4$$

The eigenvalues of A:

```
> fnormal(Eigenvalues(A))
```

$$\begin{bmatrix} -1.807392228 + 0.I \\ 1.807392228 + 0.I \\ 0.I \\ 0.I \end{bmatrix}$$

The poles of the transfer function are:

```
> fsolve(denom(tf[2,1]),s);
```

$$-1.807392228, 0., 0., 1.807392228$$

The eigenvalues are equal to the poles of the transfer function since the system is controllable and observable. Since there is an eigenvalue in the left half plane, the system is unstable.

LQR Controller Design

Specify the weights on the input variables.

```
> Q := DiagonalMatrix([1,0,1,0]) :
  R := IdentityMatrix(1) :
```

Note: The following command uses the optional Control Design Toolbox. The output of this command is copied below so that you can use this application without the toolbox.

```
> #K:=ControlDesign:-LQR(sysLin[1],Q,R)
```

$$K := \left[\begin{array}{cccc} 84.4997540652218 & 147.768218117330 & -1. & -6.02607321694750 \end{array}\right]:$$

Hence, the controlled input is:

> controller := u(t) = (-K • z)[1]

$$controller := u(t) = -84.4997540652218\,\phi(t) - 147.768218117330\,\theta(t) + 1.x(t) + 6.02607321694750\,y(t)$$

Simulate the Controlled System

Substitute the controller into the system equations.

> sysEqsController := eval(sysEqs, controller)

$$sysEqsController := \left[\left[\dot{x}(t) = y(t)\right],\ \left[\dot{\theta}(t) = \phi(t)\right],\ \left[\dot{y}(t) = -\frac{1}{-3\cos(\theta(t))\,m + 2M + 2m}\left(2L\,\phi(t)^2\sin(\theta(t))\,m - 3\cos(\theta(t))\sin(\theta(t))\,g + 168.9995081\,\phi(t) + 295.5364362\,\theta(t) - 2.x(t) - 12.05214643\,y(t)\right)\right],\ \left[\dot{\phi}(t) = -\frac{1}{(-3\cos(\theta(t))\,m + 2M + 2m)\,mL}\left(3\left(L\,\phi(t)^2\sin(\theta(t))\,m^2 - \sin(\theta(t))\,gM - \sin(\theta(t))\,gm - (-84.4997540652218\,\phi(t) - 147.768218117330\,\theta(t) + 1.x(t) + 6.02607321694750\,y(t))\,m\right)\right)\right]$$

The initial conditions (note that the angle of the pendulum is not vertical):

> ics := x(0) = 0, θ(0) = 0.25π, y(0) = 0, φ(0) = 0:

> res := dsolve({sysEqsController[1], sysEqsController[2], sysEqsController[3], sysEqsController[4], ics}, {φ(t), θ(t), x(t), y(t)}, type = numeric, output = listprocedure):

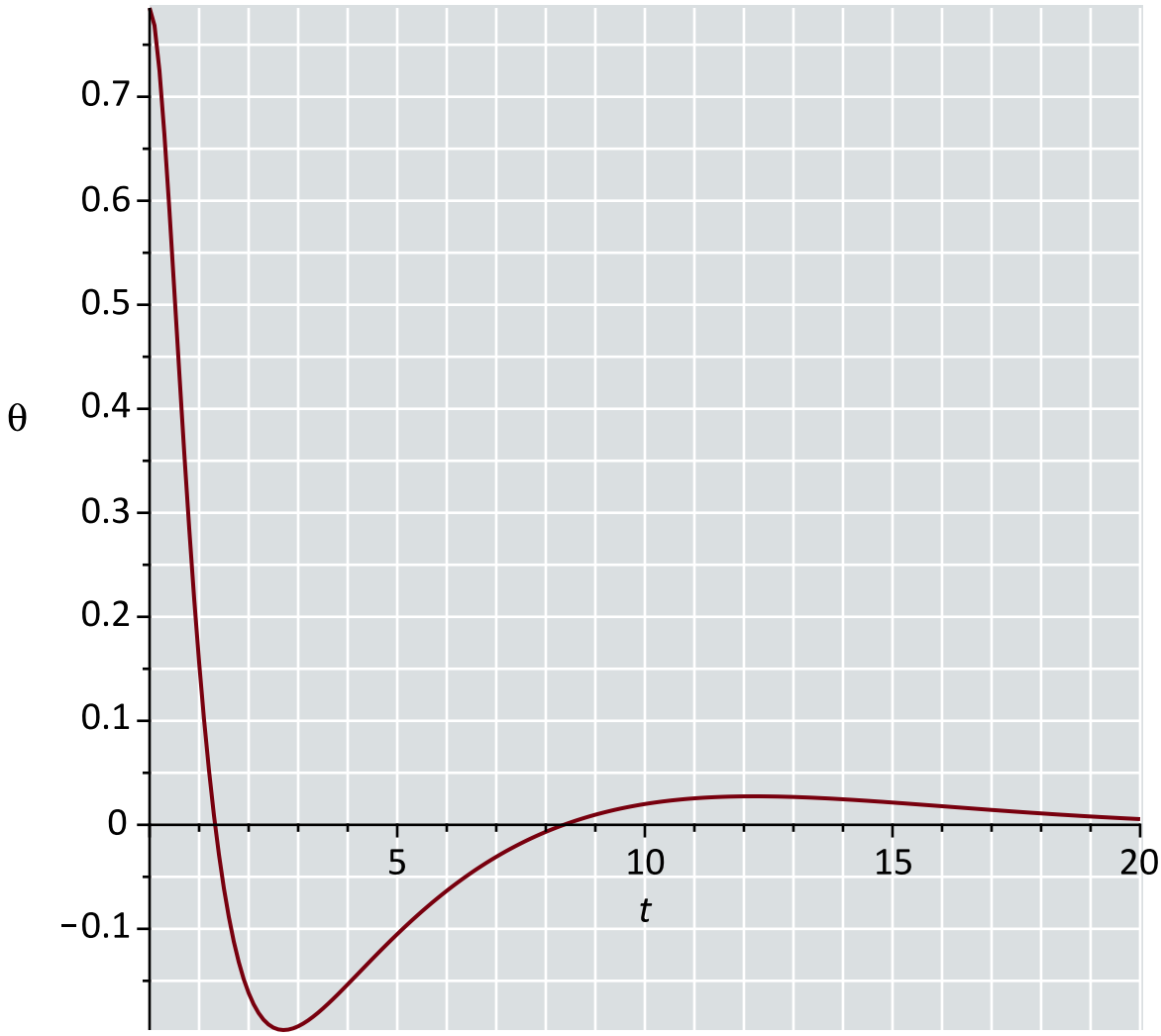
> res_x := eval(x(t), res);
res_theta := eval(θ(t), res)

$$res\_x := \mathbf{proc}(t)\ \ldots\ \mathbf{end\ proc}$$

$$res\_theta := \mathbf{proc}(t)\ \ldots\ \mathbf{end\ proc}$$

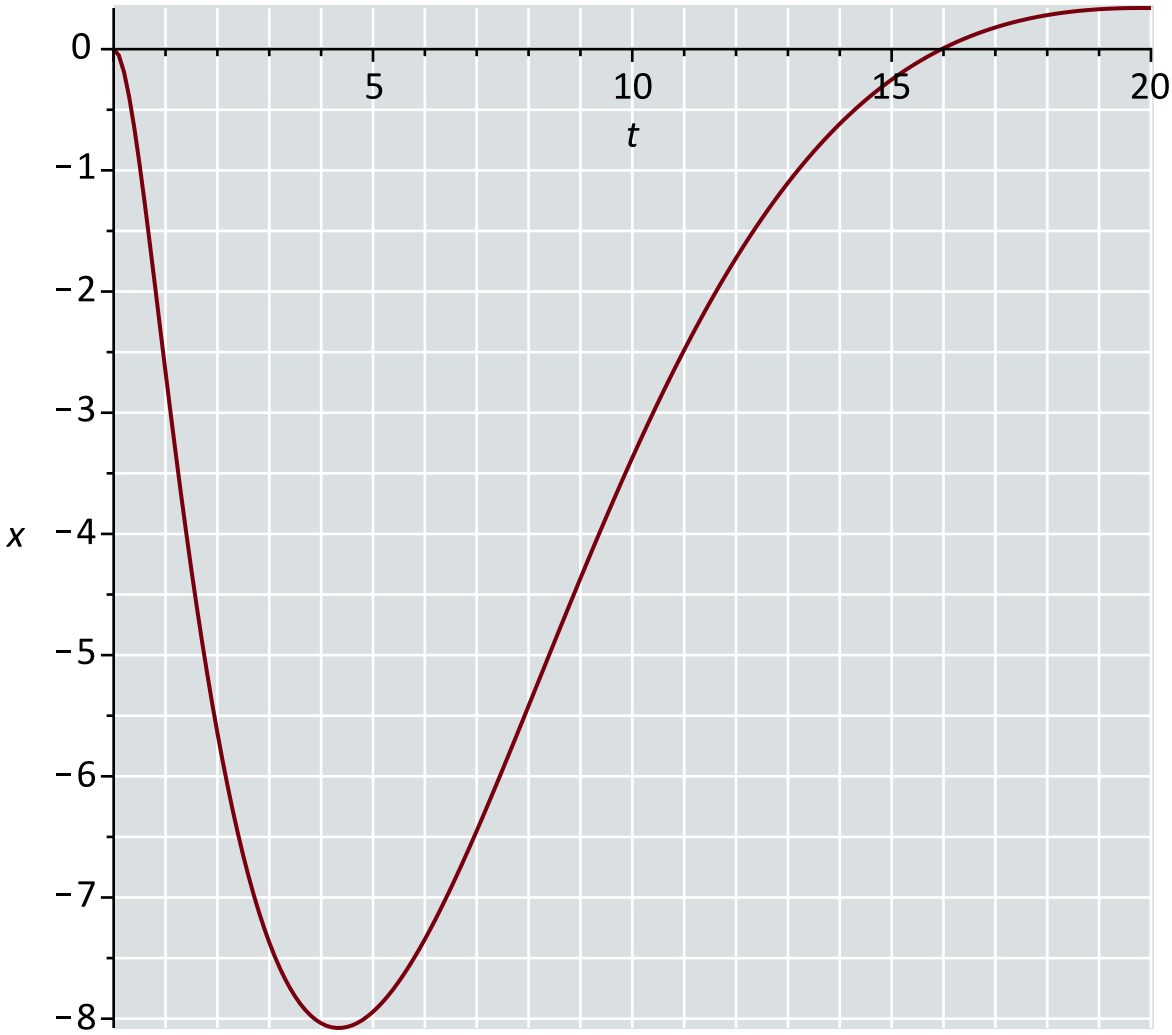
The angular displacement of the pendulum with respect to time.

> plots:-odeplot(res, [t, θ(t)], 0..20, background = ColorTools:-Color("RGB", [218/255, 223/255, 225/255]), axis = [gridlines = [color = RGB(1, 1, 1)]], size = [800, 400], *axesfont* = [*Calibri*], *labelfont* = [*Calibri*])



The cart displacement with respect to time.

```
> plots:-odeplot( res, [ t, x( t ) ], 0 ..20, background = ColorTools:-Color( "RGB", [ 218 / 255, 223
  / 255, 225 / 255 ] ), axis = [ gridlines = [ color = RGB( 1, 1, 1 ) ] ], size = [ 800, 400 ], axesfont
  = [ Calibri ], labelfont = [ Calibri ] )
```



The pendulum returns to its upright position and the cart returns to its initial position, as intended by the design of the controller.

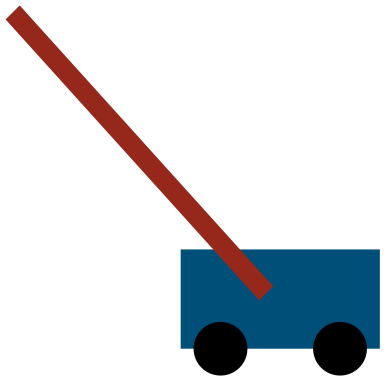
Animation of Controlled Cart

```
> pendplot := proc(x, t)
    uses plottools, plots, ColorTools :
    local pt1, pt2, pt3, pt4, pendulum, wheel1, wheel2, cart, w, r1, r2, y0;
    cart := rectangle([x - 1, 1], [x + 1, 0], color = Color("RGB", [0/255, 79/255, 121
    /255]), style = surface);
    wheel1 := disk([x - 0.6, 0], 0.25, color = black);
    wheel2 := disk([x + 0.6, 0], 0.25, color = black);
    w := 0.1;
    r1 := 3.8;
    r2 := 0.2;
    y0 := 0.7;
    pt1 := [x - r1·sin(t) + w·cos(t), y0 + r1·cos(t) + w·sin(t)];
    pt2 := [x - r1·sin(t) - w·cos(t), y0 + r1·cos(t) - w·sin(t)];
```

```
    pt3 := [x - r2·sin(t) - w·cos(t), y0 - r2·cos(t) - w·sin(t)];
    pt4 := [x - r2·sin(t) + w·cos(t), y0 - r2·cos(t) + w·sin(t)];
    pendulum := polygon([pt1, pt2, pt3, pt4], color = Color("RGB", [150/255, 40/255, 27
    /255]), style = surface);
    display(pendulum, wheel1, wheel2, cart)
end proc:

> plots:-animate(pendplot, [res_x(t), res_theta(t)], t = 0..25, frames = 100, scaling
    = constrained, axes = none)
```

$$t = 0.$$



```
>
```