Belt drive simulation

March 16, 2023

THREAD
EUROPEAN TRAINING NETWORK

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 860124.

Table 1: Main parameters for the belt drive.

Parameter	Value	Units	Description
r	0.09995	m	pulley radius
d	0.1π	m	distance between two pulleys
h_b	0.0001	m	belt height
w_b	0.08	m	belt width
$\bar{l}_b$	0.38π	m	stress-free belt length
l_b	0.4π	m	initial, deformed belt length
ε_{ref}	-0.05	-	added reference axial strain of the belt
u_0	0.	m	horizontal displacement (used only in original model [3])
EA	8000	N m	axial stiffness
EI	$\frac{4}{3} \cdot 10^{-3}$	N m^2	bending stiffness
ρ	1036	kg/m^3	beam density
dEA	1	N/ms^2	strain proportional damping
ω_{P1}	12	rad s^{-1}	angular velocity of driving pulley
d_{P2}	2	Nm/s	angular velocity proportional damping at pulley P_2
t_0	0.05	s	driving start time
t_1	0.60	s	driving end time
$t_{\tau 0}$	1.0	s	torque τ_{P2} raised at pulley P_2
$t_{\tau 1}$	1.5	s	torque τ_{P2} at pulley P_2 reaches nominal value
I_p	0.25	kg m^{-2}	moment of inertia of pulleys
g	9.81	m s^{-2}	gravity

Contents

1 Description of belt drive simulation

The belt drive has two pulleys P_1 and P_2 with identical radius and inertia, see the geometrical setup in Figure 1. The numerical modeling of the belt was based on the Absolute Nodal Coordinate Formulation (ANCF), [1]. The pulleys were simulated as rigid bodies while the contact between belt and pulleys was modeled as described in [2].

This numerical example is similar to the one developed in [3] with some modifications which attempt to eliminate the vibrations in the beginning of the simulation and allow the system to reach the steady state. The angular velocity of pulley P_1 is prescribed by means of an algebraic constraint, while some resistance torque over time is added to pulley P_2, see the description hereafter. The

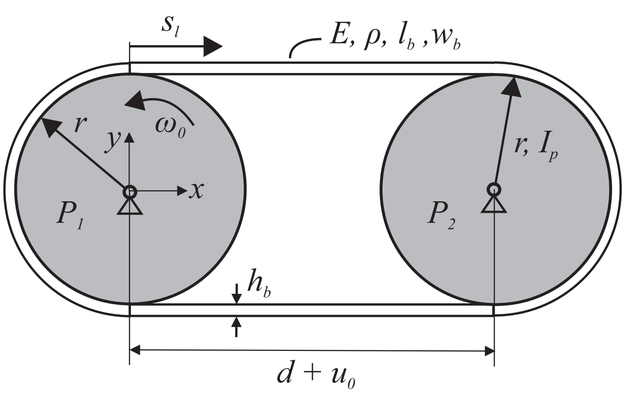

Figure 1: Belt drive with two pulleys, displaced from initial position by u_0.

belt is modelled as Bernoulli-Euler beam with bending stiffness EI, axial stiffness EA, rectangular cross section with height h_b and width w_b, as well as stretch proportional damping, density, and further parameters given in Table 1. A constant acceleration is prescribed to pulley P_1 between t_0 and t_1:

$$\omega_{P1}(t) = \begin{cases} 0\,\frac{\mathrm{rad}}{\mathrm{s}}, & \text{if} \quad t < t_0 \\ \omega_{P1}\frac{t-t_0}{t_0-t_1} & \text{if} \quad t_0 < t < t_1 \\ 12 & \text{else}\,. \end{cases} \tag{1}$$

A torque proportional to the angular velocity is applied to the pulley P_2 which represents damping of rotational motion:

$$\tau_{P2}(t) = \begin{cases} 0\,\mathrm{Nm}, & \text{if} \quad t < 1 \\ 25\,(0.5 - 0.5 \cdot \cos(2(t-1)\pi))\,\mathrm{Nm} & \text{if} \quad 1 < t < 1.5 \\ 25\,\mathrm{Nm} & \text{else}\,. \end{cases} \tag{2}$$

As compared to [3], we use a much smaller belt height h_b in order to exclude bending effects, a higher pre-tension (due to pre-stretch), while keeping the axial stiffness EA the same. Furthermore, the bending stiffness is lowered by a factor of 50, which reduces bending effects, as it would lead to significant deviations from an analytical solution otherwise. The support of pulley P_1 is not displaced during the first 0.05 s of the simulation, but the pre-stretch ε_{ref} is applied before running a static computation, which defines a static equilibrium for the dynamic simulation hereafter. The contact stiffness, as shown later by the nominal simulation parameters, has been increased by a factor of 40 and a tangential stiffness (bristle) model has been included in order to retrieve highly accurate contact behavior.

2 Description of code

For simulating the system we are using the multibody dynamics code Exudyn [4]. The code in divided into sections (1, 2,. . . , 8) and subsections (A, B, . . .) for easier documenting and processing, see section 3 with subsections (A, B, ..., E):

- In the first section we import necessary modules.
- The third section consists of the Parameter Function. This function will be repeatedly called from Parameter Variation to update the value of the variables for which we perform variations.
- In subsection 3.A we create a class P which contains all parameters for which we can perform Parameter variations. First the parameters are given their default values. Then through:

  ```
  for key,value in parameterSet.items():
      setattr(P,key,value)
  ```

  we updated the varying parameters.
- In subsection 3.B we create the model which is going to be updated for every parameter variation.
- For the ANCF beam elements modeling the belt we are using ObjectANCFCable2D, see the documentation of Exudyn[1], theDoc. Parameters used for ObjectANCFCable2D are defined in Table 2.
- For increasing the angular velocity, we are using a user function

```
def UFvelocityDrive(mbs, t, itemNumber, lOffset):
    if t < tAccStart:  # driving start time
        v = 0
    if t >= tAccStart and t < tAccEnd:
        v = omegaFinal/(tAccEnd-tAccStart)*(t-tAccStart)
    elif t >= tAccEnd:
        v = omegaFinal
    return v
```

(The file is still under construction.)

[1]EXUDYN V1.3.86.dev1; https://github.com/jgerstmayr/EXUDYN

Table 2: Parameters for ObjectANCFCable2D

Parameter	Value
physicsMassPerLength	ρA
physicsBendingStiffness	EI
physicsAxialStiffness	EA
physicsBendingDamping	dEI
physicsAxialDamping	dEA
physicsReferenceAxialStrain	ε_{ref}
physicsReferenceCurvature	0
useReducedOrderIntegration	2
strainIsRelativeToReference	False
visualization	(to be added...)

3 Installation and running

3.1 Installing python and Exudyn

The code was tested in a Windows pc using Anaconda, 64bit, Python 3.7.6 and Spyder 4.0.1 which is included in the Anaconda installation.

For installing Exudyn PIP INSTALLER (pypi.org) was used based on the following instructions: Pre-built versions of Exudyn are hosted on `pypi.org`, see the project

- https://pypi.org/project/exudyn

As with most other packages, in the regular case (if your binary has been pre-built) you just need to do[2]

```
pip install exudyn
```

On Linux (currently only pre-built for UBUNTU, but should work on many other linux platforms), **update pip to at least 20.3** and use

```
pip3 install exudyn
```

For pre-releases (use with care!), add '−−pre' flag:

```
pip install exudyn −−pre
```

For more information for installing Exudyn see the theDoc[3].

3.2 Running the code

In section 7 of the code one can choose between performing single simulation and performing parameter variation. An option for plotting figures is given. Solutions for parameters variations were stored in solution folder. Solutions from new runs are stored by default in solutionNosync.

(The file is still under construction.)

References

[1] J. Gerstmayr and H. Irschik, "On the correct representation of bending and axial deformation in the absolute nodal coordinate formulation with an elastic line approach," *Journal of Sound and Vibration*, vol. 310, no. 3, pp. 461–487, 2008.

[2] K. Ntarladima, M. Pieber, and J. Gerstmayr, "Contact modeling between axially moving beams and sheaves," under submission.

[3] A. Pechstein and J. Gerstmayr, "A Lagrange-Eulerian formulation of an axially moving beam based on the absolute nodal coordinate formulation," *Multibody System Dynamics*, vol. 30, no. 3, pp. 343–358, 2013.

[4] J. Gerstmayr, "Exudyn – A C++ based Python package for flexible multibody systems," in *the proceedings of the 6th Joint International Conference on Multibody System Dynamics and the 10th Asian Conference on Multibody System Dynamics*, (New Delhi, India), 2022, submitted.

[2]If the index of pypi is not updated, it may help to use `pip install -i https://pypi.org/project/ exudyn`

[3]EXUDYN V1.3.86.dev1; https://github.com/jgerstmayr/EXUDYN