DigiPyRo Project
Basic Examples with DigiPyRo

Samuel May

July 13, 2016

1 Creating and Digitally Rotating a Synthetic Movie

First, we will walk through the steps to create a synthetic movie of a ball in free motion on a parabolic table. Then we will digitally rotate our synthetic movie to reveal inertial circles, the canonical response of a particle in free motion in a rotating system. This instructable assumes that you have already downloaded and installed DigiPyRo (if not, go ahead and follow the instructable for doing that!).

1. Set the parameters for the synthetic movie. First, open `synths.py`. This can be done with a text editor like `vim` (my personal choice) or `emacs`, or by using an IDE like `PyCharm`. You should see something like this:
   There are 9 parameters which can be changed by the user:

   (a) The length of the movie (in seconds). In general, you'll want at least 5 seconds to get good results with `DigiPyRo`. Make this shorter to decrease computation time.

   (b) The frame rate of the movie (in frames/second). 30 (the default value) is a good value to start with, but you may want to decrease this if computation time is an issue (try 10).

   (c) Resolution of the movie (in pixels). 1260x780 (the default values) are good to start with, but again, you may want to decrease this if computation time is an issue (try 630x360).

   (d) Frequency of oscillations. The default value (10 rpm) is a good starting point, but it may be interesting to play around with this value to see how it changes the resulting inertial circles (which come after `DigiPyRo`-ing).

   (e) The rotation rate of the reference frame. The two natural frames of reference are the non-rotating frame (0) and the co-rotating reference frame (`rotRate = rpm`).

```python
# This program creates a synthetic .avi movie for use with DigiPyRo
# The video shows a ball rolling on a parabolic surface
# The user may change the length of the movie[1], the frame rate of the movie[2], the resolution of the movie[3]
# the frequency of oscillations[4], the rotation rate of the reference frame[5]
# and control the initial conditions of the roll [6]-[9]

# Import necessary modules
import cv2
import numpy as np
from Tkinter import *
import matplotlib
matplotlib.use("Agg")
from matplotlib import pyplot as plt
import scipy as sp
from scipy.optimize import leastsq

# Ask user for movie name
saveFile = raw_input('Enter a name for the movie (e.g. mySyntheticMovie): ')
saveFile += '.avi'

# Define movie details
movLength = 2                        # [1] define the desired length of the movie in seconds
fps = 30.0                           # [2] Set this to a low value (10-15) for increased speed or a higher value (30-60) for
width = 1260                         # [3] Width and height in pixels
height = 720                         # [3] Decrease the width and height for increased speed, increase for improved resoluti

# Define table values
rpm = 10.0                           # [4] frequency of oscillations (in RPM). Good values might be 5-15
                                     # NOTE: A two-dimensional rotating system naturally takes the shape of a parabola.
                                     # The rotation rate determines the curvature of the parabola, which is why we define th
rotRate = 10.0                       # [5] rotation rate of camera. The two natural frames of reference are with rotRate = 0


# Set initial conditions
r0 = 1.0                             # [6] initial radial position of ball. Choose a value between 0 and 1
vr0 = 0.0                            # [7] initial radial velocity of ball. Good values might be 0-1
phi0 = np.pi/4                       # [8] initial azimuthal position of ball. Choose a value between 0 and 2*pi
vphi0 = 0.0                          # [9] initial azimuthal velocity of ball. Good values might be 0-1
```

(f) The initial radial position of the ball. This is defined in terms of a fraction of the radius of the circle that is inscribed inside the rectangular movie frame. This should always be a value between 0 and 1.

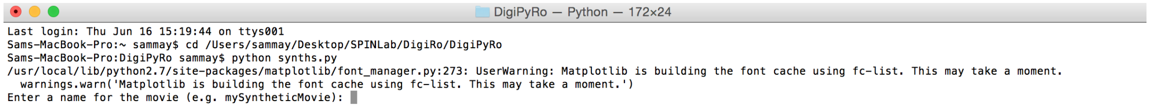
(g) The initial radial velocity of the ball. Setting this to 0 simulates a roll from rest. Try other values, perhaps between -1 and 1 to see the effects of an initial radial velocity.

(h) The initial azimuthal position of the ball. Choose a value between 0 and 2π.

(i) The initial azimuthal velocity of the ball. Setting this to 0 simulates a roll from rest. Try other values, perhaps between -1 and 1 to see the effects of an initial azimuthal velocity.

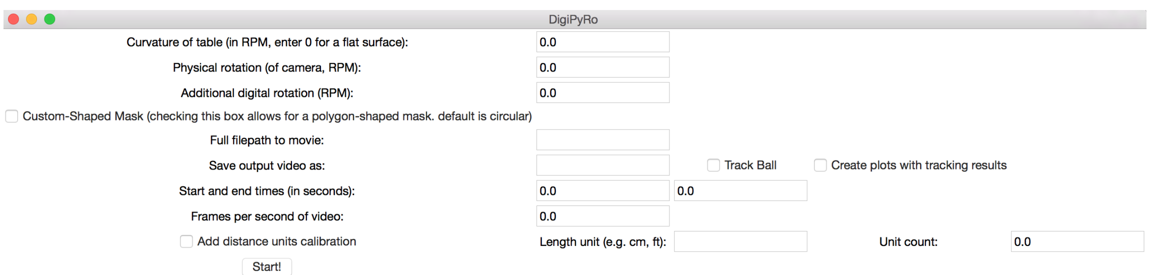
2. Create the synthetic movie. Open a terminal and navigate to your `DigiPyRo` directory, then run `synths.py`:
`cd /`*insert_path_to_your_DigiPyRo_directory_here*`/DigiPyRo`

`python synths.py`
You'll be prompted to enter a name for the movie:

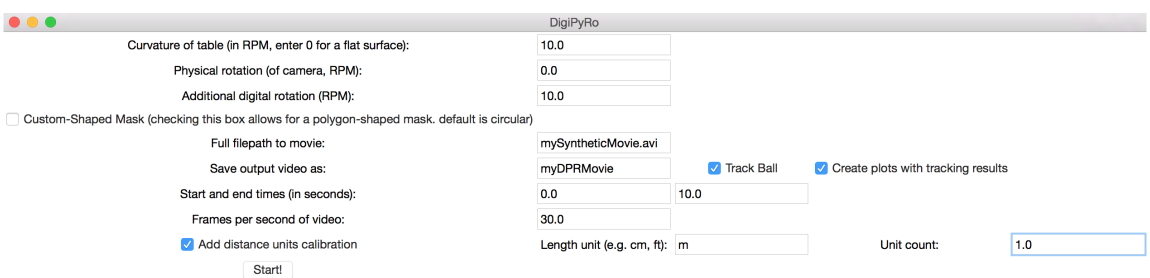


enter your desired name for the movie *without* a file extension – it will automatically be saved as a `.avi` file. For example if I wanted to name the movie "mySyntheticMovie", I would enter:
`mySyntheticMovie`
then press `return`.
By default, the movie will be saved in your `DigiPyRo` directory. If you wish to manually specify a different destination, supply the path to that destination along with the movie name. For example:
`/Users/myname/Documents/mySyntheticMovie`

3. `DigiPyRo` the synthetic movie. Open a terminal, navigate to your `DigiPyRo` directory, then run `DigiPyRo`:
`cd /`*insert_path_to_your_DigiPyRo_directory_here*`/DigiPyRo`
`python DigiPyRo.py`
you should see a menu like this: Assuming we are using a synthetic movie created



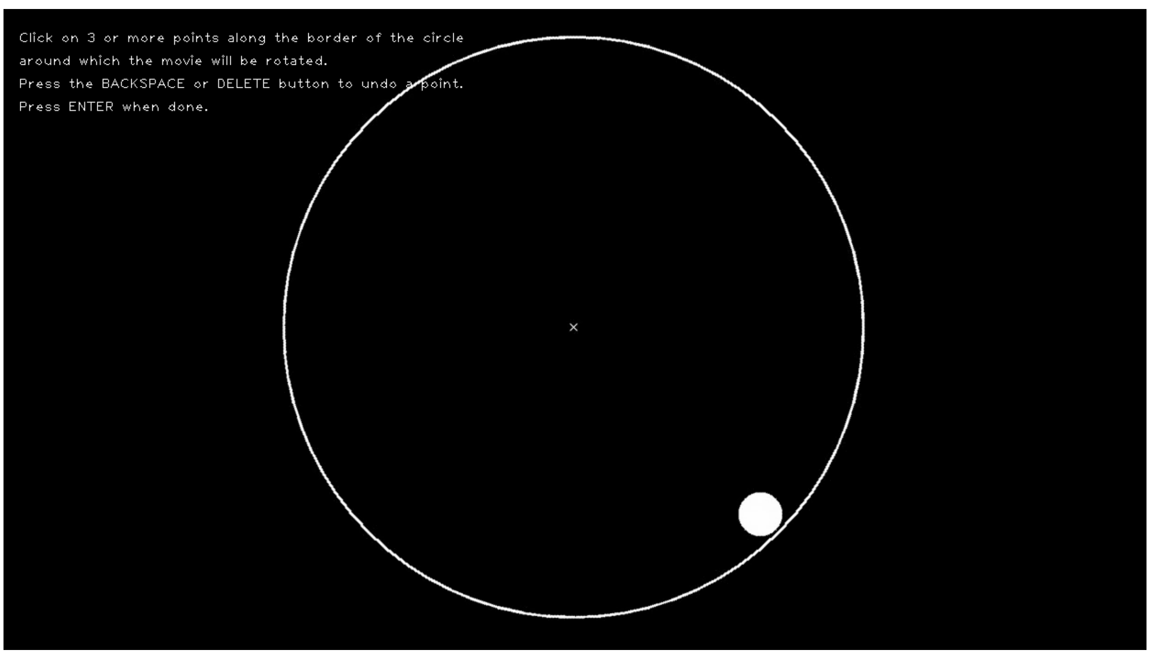
with the default parameters, the menu should be filled out like this: A few things to



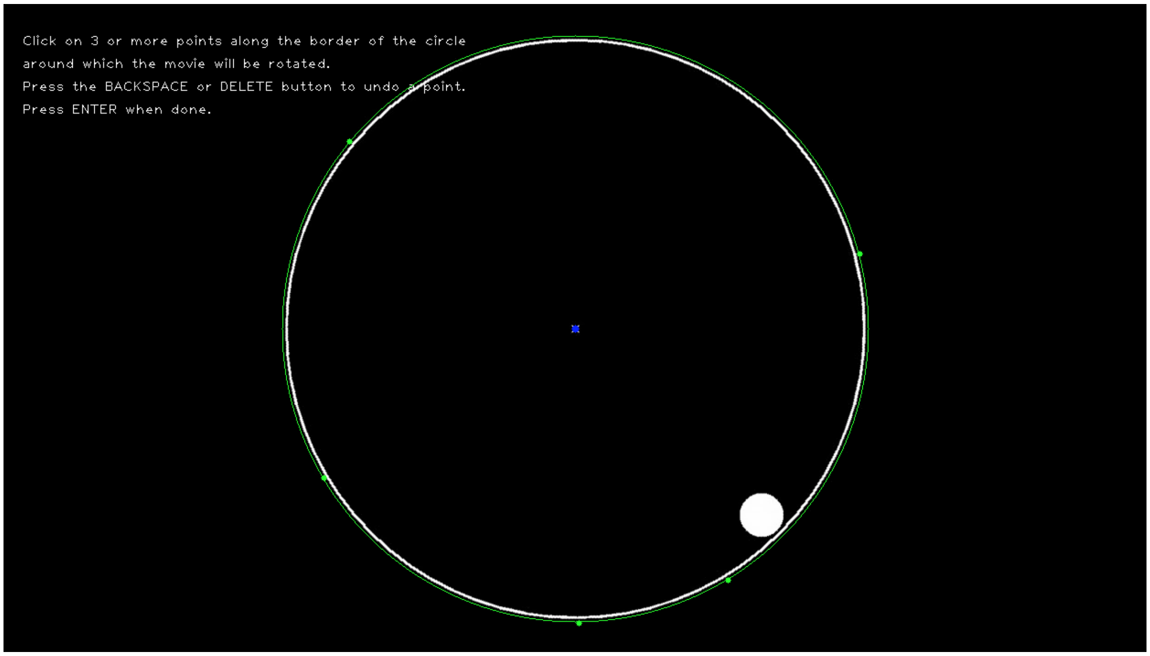
note:

- `Full filepath to movie:` if you are not selecting a video that is already in your `DigiPyRo` directory, you must specify the full path to the movie. You must also specify the extension of the movie (i.e. `.avi`, `.mp4`, etc.).
- Again, follow the naming conventions for creating the synthetic movie when creating your `DigiPyRo`-ed movie. This means that you should not specify a file extension and that the movie will be saved in your `DigiPyRo` directory unless you manually specify a different path.
- If you would like to track the ball, select the checkbox. This will also allow `DigiPyRo` to create an estimate of the inertial radius. Checking this option also means a `.txt` file will be created with the tracking data.
- If you would like plots of the tracking data, select the checkbox. This should only be checked if particle-tracking is selected.
- If you would like to express the tracking data in units of length rather than in pixels, select the `Add distance units calibration` checkbox and specify your unit of choice. You will be asked to draw a line of `unit count` units. So if I placed a 12 inch ruler in my movie and wanted to use that to convert my units to inches, I would enter “in” as my `length unit` and “12” as the `unit count`.

Once you have finished selecting your options, press the `Start!` button. You should be prompted with a window like this: click 3 or more points along the border of the

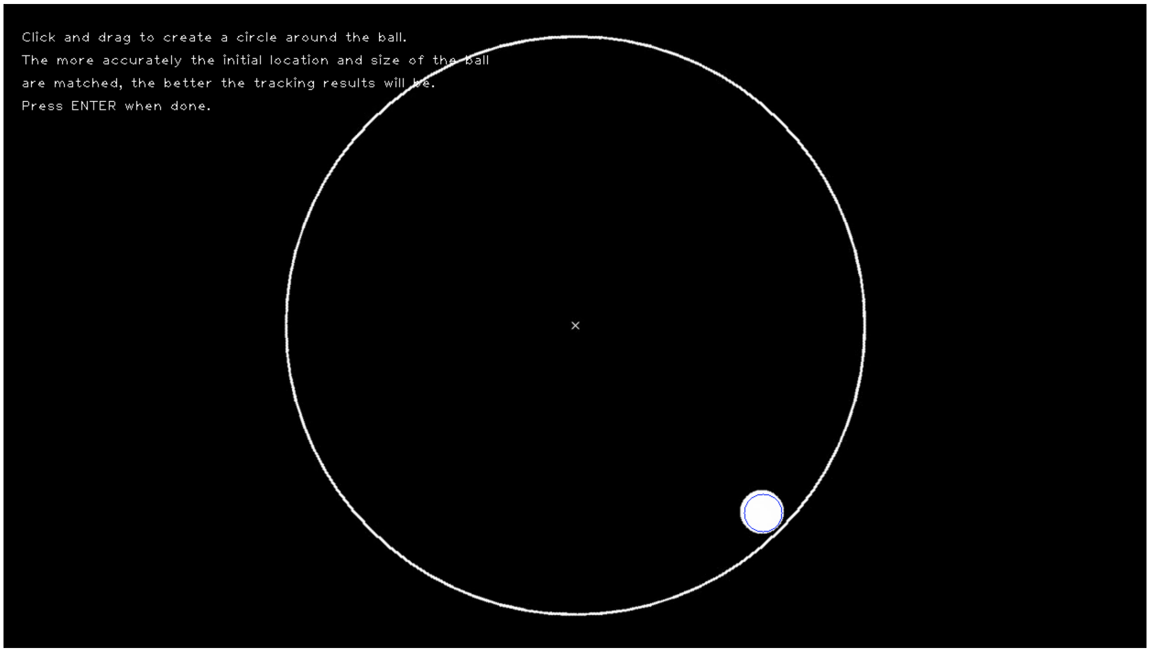


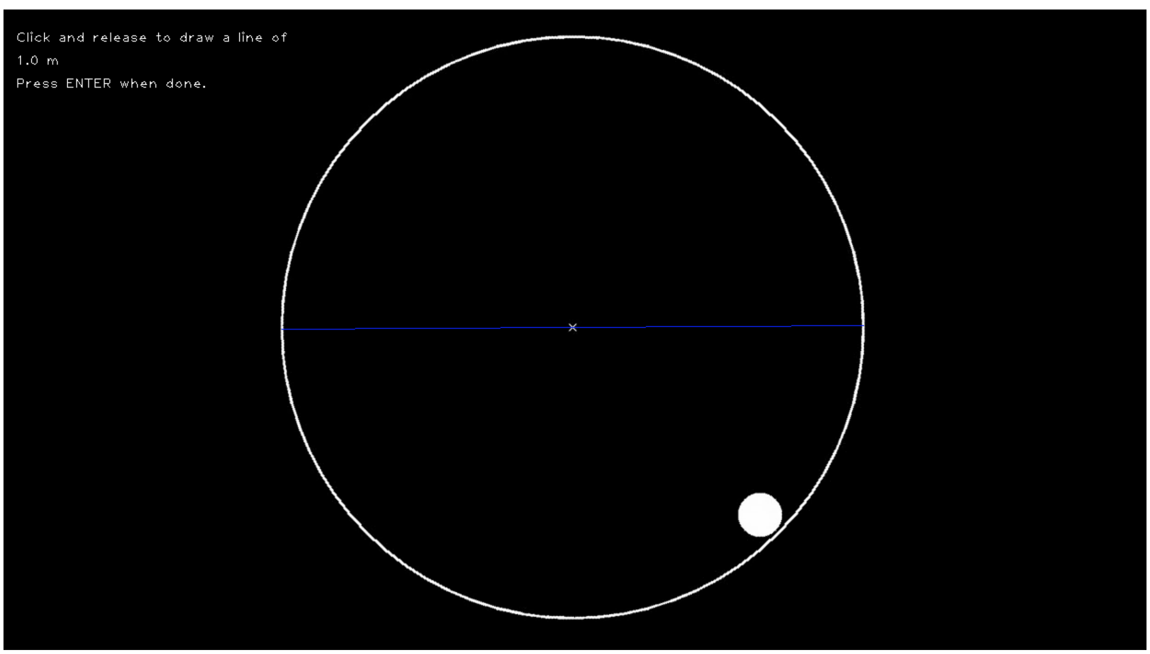
circle in order to identify the mask region (`DigiPyRo` will black out all parts of the video outside the region of interest) and the point of rotation. After doing that, your screen should look something like this: Press enter when done.

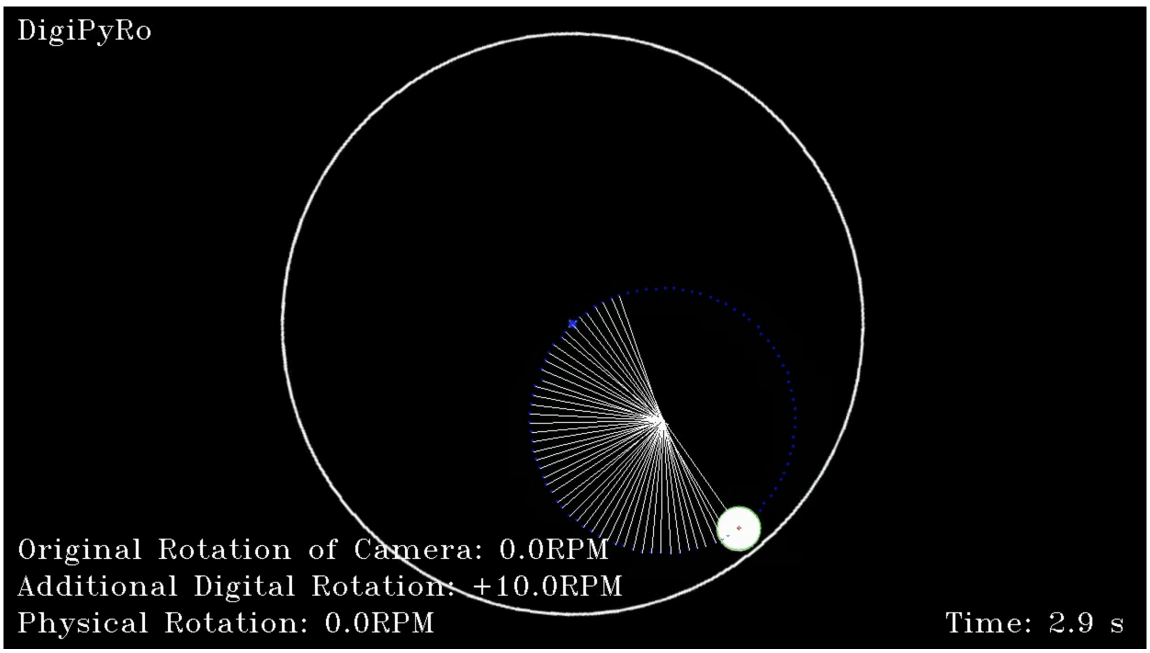


Next, you will be prompted to draw a circle around the ball. Simply click down on one end of the ball, drag across, and release the mouse. The circle you have drawn will be shown in blue. Redraw the circle as many times as necessary. IT does not need to be perfect, as the circle simply gives `DigiPyRo` an approximate size and location in which to search for circular objects. You should end up with something like this: Again, press enter when done. Now wait for `DigiPyRo` to digitally rotate your movie! It may take 1-10 seconds of computation time for each second of video time depending on frame rate, movie resolution, and the speed of your computer.

Next, you will be prompted to draw a line for length units calibration. Let's assert that the diameter of the circle in our video is 1 meter. We are prompted to click and release to draw a line of 1.0 m – the resulting line is shown in blue: A snapshot from the `DigiPyRo`-ed movie is shown below: Also shown are examples of the `.txt` file and the plots of tracking data.


Click and drag to create a circle around the ball.
The more accurately the initial location and size of the ball
are matched, the better the tracking results will be.
Press ENTER when done.


Click and release to draw a line of
1.0 m
Press ENTER when done.

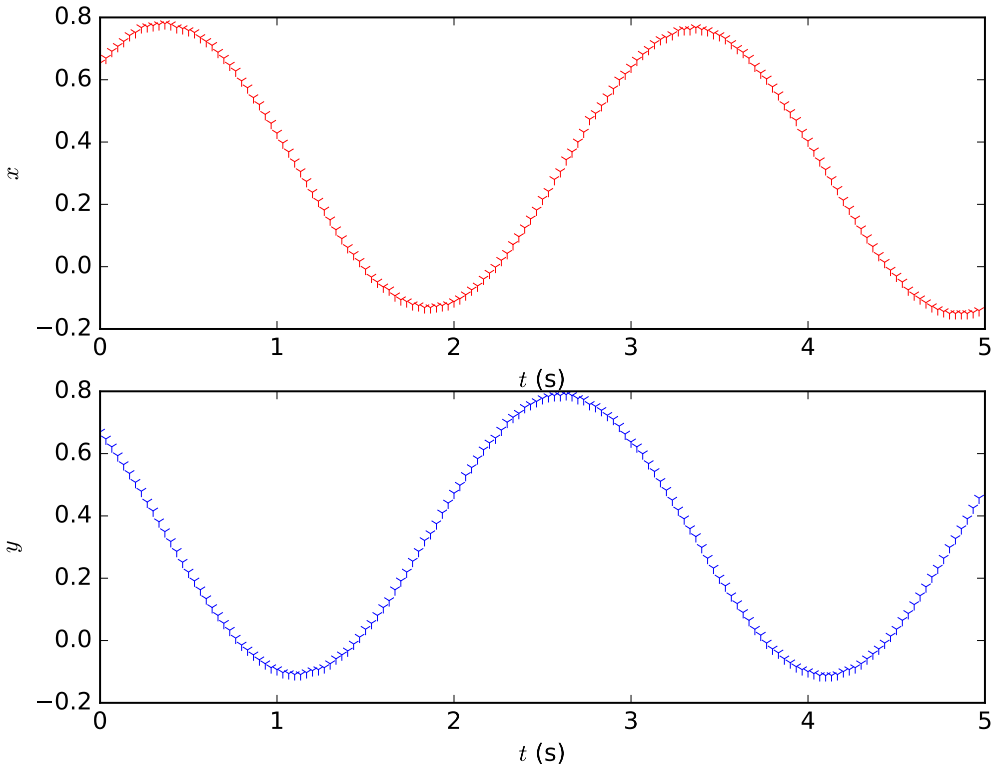

DigiPyRo
Original Rotation of Camera: 0.0RPM
Additional Digital Rotation: +10.0RPM
Physical Rotation: 0.0RPM
Time: 2.9 s

```
DigiPyRo Run Details

Original File: testin.avi
Output File: test_6.30.16
Date of run: Thu Jun 30 15:42:57 2016

Original rotation of camera: 0.0 RPM
Added digital rotation: 10.0 RPM
Curvature of surface: 10.0 RPM

Particle Tracking Data in m and m/s
t x y r theta u_x u_y u_r u_theta ||u||
0.00 0.65 0.67 0.94 0.80 0.47 -0.66 -0.13 -0.86 0.82
0.03 0.67 0.65 0.93 0.77 0.57 -0.71 -0.08 -0.97 0.91
0.07 0.69 0.62 0.93 0.73 0.57 -0.81 -0.12 -1.06 0.99
0.10 0.71 0.59 0.92 0.70 0.47 -0.85 -0.19 -1.04 0.98
0.13 0.72 0.56 0.92 0.66 0.52 -0.85 -0.11 -1.08 1.00
0.17 0.74 0.54 0.92 0.63 0.43 -0.85 -0.16 -1.03 0.96
0.20 0.75 0.51 0.91 0.59 0.38 -0.85 -0.16 -1.01 0.93
0.23 0.77 0.48 0.90 0.56 0.28 -0.95 -0.26 -1.06 0.99
0.27 0.77 0.44 0.89 0.52 0.09 -0.95 -0.39 -0.97 0.95
0.30 0.77 0.42 0.88 0.49 0.09 -0.95 -0.36 -1.01 0.95
0.33 0.78 0.38 0.87 0.46 0.09 -1.00 -0.36 -1.08 1.00
0.37 0.78 0.35 0.85 0.42 0.00 -0.95 -0.39 -1.02 0.95
0.40 0.78 0.32 0.84 0.39 -0.19 -0.95 -0.54 -0.96 0.97
0.43 0.77 0.29 0.82 0.36 -0.19 -1.00 -0.52 -1.06 1.01
0.47 0.76 0.25 0.80 0.32 -0.14 -1.00 -0.45 -1.12 1.01
0.50 0.76 0.22 0.79 0.28 -0.19 -0.90 -0.44 -1.03 0.92
0.53 0.75 0.19 0.78 0.25 -0.33 -0.85 -0.53 -0.97 0.92
```



0.8
0.6
0.4
0.2
0.0
−0.2
x
0
1
2
3
4
5
t (s)
0.8
0.6
0.4
0.2
0.0
−0.2
y
0
1
2
3
4
5
t (s)