# Kinematic Constraints vs Multibody Dynamics

### Anne Koelewijn, Cleveland State University

### January 25, 2017

Posa et al. recently described a method for trajectory optimization with kinematic constraints [1]. This method was originally developed to handle multibody systems with kinematic loops. We were, however curious, if we should consider formulating the entire system this way. This would eliminate the need for multibody dynamics.

In order to answer this question, we compared the conventional approach for solving a trajectory optimization problem and the approach by Posa et al. [1], which will be called the kinematic constraint approach. Both approaches are applied to several problems on a one DOF pendulum with varying duration and target angle.

The pendulum is shown in figure 1. It is operated by a torque at the base of the pendulum and the angle is defined as shown in the picture. It has a massless rod with length and mass .

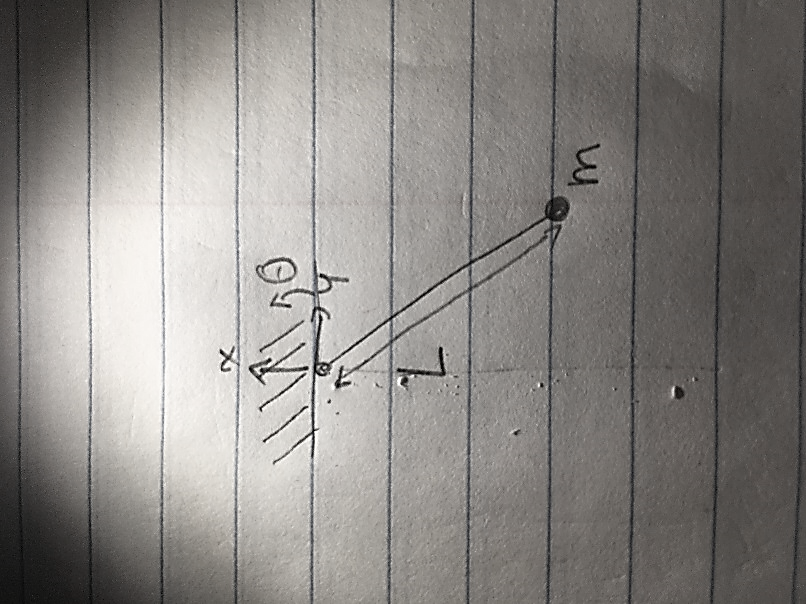


Figure - Pendulum

## Objective

The objective is to move from the downward position to the target angle using minimum energy. Therefore, the following objective is used:

|  |  |  |
| --- | --- | --- |
|  |  | 1 |

## Constraints – Conventional Approach

In the conventional multibody dynamics approach, the following equations of motion, , are used:

|  |  |  |
| --- | --- | --- |
|  |  | 2 |

where denotes the angular velocity. The mass and length were 1 kg and 1 m. The dynamics constraints are as follows:

|  |  |  |
| --- | --- | --- |
|  |  | 3 |

The task constraints are as follows:

|  |  |  |
| --- | --- | --- |
|  |  | 4 |

where is the target angle, measured from the starting position.

An optimal control problem is formulated as follows:

**find a trajectory , , that satisfies the dynamic constraints (3), the task constraints (4), and minimizes the cost function in equation 1.**

The problem was solved using direct collocation. The trajectory was discretized into time nodes with a time step of 0.01 s (for instance 100 nodes for 1 second) and the dynamics were approximated by a backward Euler (BE) discretization. The resulting nonlinear optimization problem was solved with IPOPT.

## Constraints – Kinematic Constraint Approach

In this approach [1], the following equations of motion, , are used:

|  |  |  |
| --- | --- | --- |
|  |  | 5 |
|  |  | 6 |

where the control input, , is the torque at the base of the rod. It is converted to a force using and . is a Lagrange multiplier representing the reaction force in the rod.

The and position are connected using the following kinematic constraint:

|  |  |  |
| --- | --- | --- |
|  |  | 7 |

This yields the following constraints on the speed and the acceleration:

|  |  |  |
| --- | --- | --- |
|  |  | 8 |
|  |  | 9 |

where equation 5 and 6 are used to determine and

The following dynamics constraints were added:

where are the equations of motion and , , , and are determined using BE discretization. are different Lagrangian multipliers than .

The following task constraints were used:

|  |  |  |
| --- | --- | --- |
|  |  | 10 |

where denotes the target angle, measured from the starting positon.

The Matlab “unwrap” function was to determine from the entire trajectory, so that final positions could be outside of the range of -180 to +180 degrees.

An optimal control problem is formulated as follows:

**find a trajectory , , , , , , that satisfies the kinematic constraints (7-9), the dynamic constraints (5, 6), and the task constraints (10), and minimizes the cost function in equation 1.**

The problem was solved using direct collocation. The trajectory was discretized into the same number of time nodes, using BE discretization. The resulting nonlinear optimization problem was solved using IPOPT.

## Problems that will be solved

A range of problems were solved on the pendulum. The initial position was always the downward position. The target angle was either 45, 90, 180 and 360 degrees from the initial position. Also, the duration varied between 0.5, 1, 2 and 4 seconds. A problem with a short time and a large motion requires a large torque, while a problem with a long time and a small motion requires very little torque, since it can use inertia. These problems are often harder to find.

## Results

The table below shows the results of the comparison between the kinematic constraint approach (kin) and the conventional approach (multi). Each problem was solved 6 times from a random initial guess. The “% solutions lowest” column indicates how many times the lowest of the 6 solutions was found. If this is 17%, it was found only once, indicating that this problem is more difficult to solve than when all 6 problem found the same solution. Then, it is likely that the solution is a global optimum. Note that sometimes, the percentage in “% solution lowest” was quite low, while the standard deviation of the objective “Std Obj” is quite small (e.g. the 90-degree task with 1 second duration using the kinematic constraint approach). This indicates that a similar solution was found each time and that it is still likely that this is the global optimum.

The conventional approach found the global optimum more often than the kinematic constraint approach. Also, in general, the average and smallest objective is lower in the conventional approach compared to the kinematic constraint approach. Finally, the solution time is around 100 times faster. There are a couple of individual cases where the kinematic constraint approach finds a better solution, but overall it is concluded that the conventional approach works a lot better.

The conventional approach especially worked better in the problems with a duration of 4 seconds, which are harder to solve. Except for the problem with the 360-degree task, a lower objective is found. Also, the average objective is lower and the number of times the global solution is found is higher in the conventional approach for all cases.

Table - Results of comparison between conventional approach (multi) and kinematic constraint approach (kin)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Angle | Duration | Sol Met | % Solutions Lowest | Average Objective | Std Obj | Smallest Objective | Average Time |
| 45 | 0.5 | kin | 100 | 84.4 | 0 | 84.4 | 37.6 |
| 45 | 0.5 | multi | 100 | 85.0 | 0 | 85.0 | 0.343 |
| 45 | 1 | kin | 83 | 11.7 | 3e-5 | 11.7 | 67.5 |
| 45 | 1 | multi | 100 | 11.2 | 0 | 11.2 | 0.682 |
| 45 | 2 | kin | 67 | 3.13 | 5e-5 | 3.13 | 197 |
| 45 | 2 | multi | 100 | 2.83 | 0 | 2.83 | 1.45 |
| 45 | 4 | kin | 17 | 101 | 117 | 0.871 | 263 |
| 45 | 4 | multi | 67 | 23.1 | 16.8 | 0.710 | 4.02 |
| 90 | 0.5 | kin | 100 | 371 | 0 | 371 | 27.4 |
| 90 | 0.5 | multi | 100 | 372 | 0 | 372 | 0.345 |
| 90 | 1 | kin | 67 | 39.1 | 3e-4 | 39.1 | 80.2 |
| 90 | 1 | multi | 100 | 37.3 | 0 | 37.3 | 0.706 |
| 90 | 2 | kin | 50 | 10.7 | 2e-4 | 10.7 | 189 |
| 90 | 2 | multi | 100 | 9.54 | 0 | 9.54 | 1.67 |
| 90 | 4 | kin | 17 | 105 | 40.2 | 60.4 | 357 |
| 90 | 4 | multi | 67 | 18.8 | 11.8 | 2.43 | 5.35 |
| 180 | 0.5 | kin | 100 | 1949 | 0 | 1949 | 27.5 |
| 180 | 0.5 | multi | 100 | 1946 | 0 | 1946 | 0.385 |
| 180 | 1 | kin | 50 | 154 | 7e-5 | 154 | 63.4 |
| 180 | 1 | multi | 100 | 148 | 0 | 148 | 0.690 |
| 180 | 2 | kin | 17 | 35.4 | 8e-3 | 35.3 | 339 |
| 180 | 2 | multi | 50 | 33.3 | 0.29 | 32.8 | 1.62 |
| 180 | 4 | kin | 17 | 22.7 | 6.75 | 7.63 | 368 |
| 180 | 4 | multi | 67 | 17.1 | 8.44 | 6.28 | 4.60 |
| 360 | 0.5 | kin | 100 | 8363 | 0 | 8363 | 34.5 |
| 360 | 0.5 | multi | 100 | 8308 | 0 | 8308 | 0.381 |
| 360 | 1 | kin | 33 | 647 | 1e-2 | 647 | 123 |
| 360 | 1 | multi | 100 | 653 | 0 | 653 | 0.728 |
| 360 | 2 | kin | 17 | 90.9 | 9e-5 | 90.9 | 168 |
| 360 | 2 | multi | 100 | 91.2 | 0 | 91.2 | 2.08 |
| 360 | 4 | kin | 17 | 118 | 84.1 | 26.1 | 466 |
| 360 | 4 | multi | 100 | 26.6 | 0 | 26.6 | 5.20 |

## Discussion

Our test results show that the conventional approach is much faster, and is also more likely to find the global optimum from a random initial guess.

A difference in computation speed was expected, but the magnitude of the speed difference was surprisingly large. This may be different when a different solver (e.g. SNOPT) is used.

The kinematic constraint method often found the same solution as the conventional method (though it took longer) but not always. Only in the 90-degree task with 2 second duration, the kinematic constraint method only found solutions that were much worse. Also, for instance in the 360-degree task with 4 second duration, the kinematic constraint method found a good solution only once, while the high standard deviation indicates that the other solutions were not good at all.

We had hoped that the kinematic constraint method would be less sensitive to the initial guess. The kinematic constraint approach has more variables and more equations, but the equations are all quadratic. There is no longer a trigonometric function in the dynamics. However, the opposite was found, the kinematic constraint method was less likely to find good solutions from a random initial guess.

The “unwrap” function that was used in the task constraint is not a differentiable function, and may have caused difficulty for the NLP solver. It may be better to introduce an additional trajectory variable (t), and an additional differential equation:

|  |  |  |
| --- | --- | --- |
|  |  | 10 |

This would do the unwrapping and allow the final position to be specified directly in terms of angle.

**References:**

[1] Posa, Michael, Scott Kuindersma, and Russ Tedrake. "Optimization and stabilization of trajectories for constrained dynamical systems." Robotics and Automation (ICRA), 2016 IEEE International Conference on. IEEE, 2016.