Partition-based Stability Controller for Push Recovery

Hyunjong Song
Department of Mechanical and Aerospace Engineering
New York University
Brooklyn, NY, USA

William Z. Peng
Department of Mechanical and Aerospace Engineering
New York University
Brooklyn, NY, USA

Joo H. Kim
Department of Mechanical and Aerospace Engineering
New York University
Brooklyn, NY, USA
Email: joo.h.kim@nyu.edu

I. Background

There is a lack of explicit criteria for balance stability within the literature. Existing stability criteria are neither accurate nor sufficient for balance [1]. Without explicit partitions of balance versus fall, existing controllers cannot exploit their systems' full balancing capability. In previous work, stability regions were proposed as partitions of augmented center-of-mass- (COM-) state space into balanced and unbalanced regions, where the unbalanced regions are further partitioned into steppable and falling regions [1]. Here, a novel controller for push recovery using the proposed partitions is introduced and validated.

II. Partition-Based Control

A stability region is a set of maximum perturbation velocities that is computed as the solutions to a series of optimization problems subject to constraints representing system properties, dynamics, environmental interactions, and partition-specific conditions (balanced or steppable) [1]. For standing push recovery, a double support standing pose with a step length of zero was considered for both balanced and steppable regions. Ankle and hip strategy balanced regions were computed by fixing all joints other than the ankle and all lower-body joints (i.e., hip, knee, and ankle), respectively.

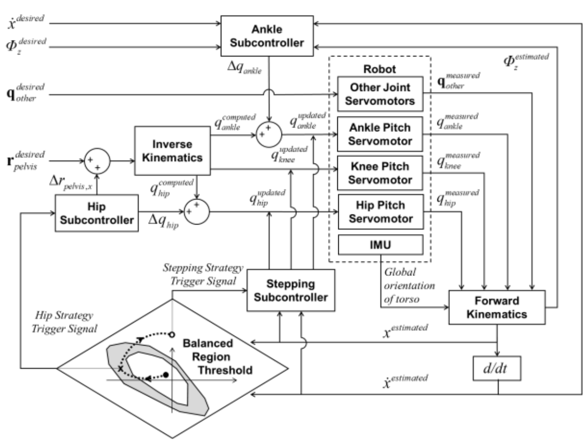

Fig. 1. Proposed partition-based push recovery controller block diagram.

The proposed partition-based controller for push recovery (Fig. 1) consists of the ankle, hip, and stepping subcontrollers. The ankle subcontroller is always engaged, regulating COM velocity and foot tipping angle with derivative control [2] and proportional control, respectively, by adjusting the ankle joint angle bias.

When the COM state exits the ankle strategy region, the hip subcontroller is triggered and generates a bang-bang-like profile for hip motion [2] by adjusting the pelvis *X*-position bias $\Delta r_{pelvis,x}$ and hip pitch angle bias Δq_{hip} to user-defined maximum values for a fixed duration and gradually to zero afterwards.

When the COM state exits the hip strategy region, the stepping subcontroller is triggered. The lower-body joints follow a pre-defined trajectory with two options: a basic stepping trajectory that achieves the given step length as quickly as possible and a capturable stepping trajectory that comes to a stop after the given step length is reached.

III. Simulation Results and Discussion

The proposed controller was tested in the Webot simulation environment with the DARwIn-OP robot model (Fig. 2). All gains and parameters were manually tuned. When testing the stepping subcontroller, falling (not steppable), steppable, and capturable velocities were considered. The capturable velocity corresponds to the capture point [3] for a step length of 0.057 m and initial COM *X*- and *Y*-positions of 0.0 m and 0.2 m, respectively. The simulation experiments demonstrated the validity of the computed stability regions, which are used actively as criteria in control without additional restrictions such as constant COM height that are used in other controllers.

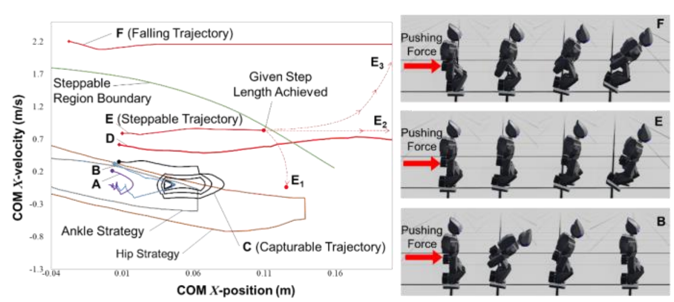

Fig. 2. Balanced (A, B, C) and unbalanced (D, E, F) trajectories from simulation results with the proposed controller. The ankle (A-F), hip (B-F), and stepping subcontrollers (E, F) selectively enabled for relative comparisons. The capturable trajectory (C) shows the robot can come to stop after making a step. The steppable trajectory (E) ends when the given step length is achieved, after which the system may either come to a stop (E_1), keep walking (E_2), or fall (E_3) depending on its subsequent actions. When the COM state is outside of steppable region (F), the robot is falling and cannot reach the given step length.

References

[1] W. Z. Peng, et al., *ASME J. Mech. Robot.*, vol. 13, no. 3, pp. 031103-1–031103-11, Jun. 2021.

[2] S. J. Yi, et al., *Proc IEEE Int. Conf. Robot. Autom.*, pp. 1649–1655, 2013.

[3] T. Koolen, et al., *Int. J. Rob. Res.*, vol. 31, no. 9, pp. 1094–1113, July 2012.