2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)
Beijing, China, November 6-9, 2018

Omni-directional Fall Avoidance of Bipedal Robots with Variable Stride Length and Step Duration

Gwanwoo Kim†, Hiroki Kuribayashi†, Yuichi Tazaki† and Yasuyoshi Yokokohji†

Abstract— **This paper proposes a capturability analysis method for fall avoidance of bipedal robots under arbitrary disturbances. Based on a dynamical model of the planar movement of the center-of-mass, capture region is computed numerically by discretizing the state space and the set of control inputs. The proposed method is able to handle a number of practically important elements of fall avoidance such as the relation between stride length and step duration, and kinematic limitations of foot placement, which have been neglected in conventional studies for simplification. The developed fall-avoidance controller utilizes precomputed capturable regions to filter reference foot placements produced by a foot-step planner to ensure fall-avoidance with small online computation time. Capture regions computed by the proposed method are compared with the conventional ones in case studies. The performance of the proposed fall-avoidance controller is evaluated in simulations.**

I. INTRODUCTION

Humanoid robots are expected to collaborate with humans in various situations such as laborious work in nursing care. Maintaining balance is one of the crucial capabilities required to humanoid robots for ensuring safety of surrounding people and robots themselves. There are three basic balancing strategies [1]. Ankle and hip strategies, which regulate ankle torque and body posture for balancing, are used to compensate small disturbance. To ensure robust fall-avoidance in the presence of large disturbances, however, stepping would be an essential strategy.

In push recovery studies, the Linear Invented Pendulum Model (LIPM) [2] is widely accepted as a simplified linear model of walking robots. In [3], Pratt et al. decomposed the LIPM dynamics into stable and unstable components and suggested the *instantaneous capture point* (ICP), which takes the same values as the *divergent component of motion* (DCM) in [4]. The ICP is such a point that if the center-of-pressure (CoP) is located on it, the center-of-mass (CoM) eventually comes to rest exactly above it. The ICP and DCM methods have been used to stabilize walking motions [5], [6].

Recent studies have proposed some solutions [7] - [15] for adjusting foot placement and/or step duration. In [7] and [8], simultaneous online planning of a CoM trajectory and foot placement are formulated as a quadratic programming (QP). However, because the step duration must be provided, it is not enough for fall-avoidance. In [9] - [13], foot placement and step duration are optimized by considering the state after only one step. Due to restrictions caused by self-collision between the legs, however, these methods are not able to find optimal foot placement and step duration when two or more steps are required to compensate disturbance. Some other works [14], [15] proposed the trajectory planning of which optimize both of the foot placement and step duration with the predetermined number of steps N. However, considering the fact that the magnitude and the direction of disturbance is not predictable, N is hard to be determined.

In [16], Koolen et al. developed the *capturability-based analysis*. The capturability-based analysis provides the definition of the set of states, which exist a sequence of control inputs able to bring the state to a balanced state, and the *N-step capture region*, which a region of the next foot placement able to bring a state to a balanced state within N steps. There are two limitations in the conventional capturability-based analysis. For simplifying the analysis, it is assumed that a robot can make every step with the maximum stride length within a fixed step duration. Under these assumptions, capturability-based analysis is essentially simplified to track the one-directional movement of the capture point. However, such simplification is not acceptable when one attempts to implement a capturability-based controller on a real humanoid robot with strict physical constraints.

In this paper, we present omni-directional capturability-based analysis that takes into consideration self-collision avoidance and the relation between stride length and step duration. Moreover, we develop a fall-avoidance controller based on N-step capture region that can be used in combination with conventional foot-step planners. It is demonstrated in simulations that a humanoid robot is able to recover balance from a large disturbance by making multiple steps.

The rest of this paper is organized as follows. In Section II, a state-space model of omni-directional stepping of a biped robot is derived. Section III briefly reviews the general caputurability analysis. In Section IV, the proposed capture region computation algorithm is presented, and in Section V, the design of a capturability-based fall-avoidance controller is described. In Section VI, the proposed method is compared with a conventional one in case studies, and the proposed fall-avoidance controller is tested in simulations. Section VII gives some concluding remarks.

II. FORMULATION OF A DYNAMICAL SYSTEM MODEL OF BIPEDAL ROBOTS

A. Linear Inverted Pendulum Model and Instantaneous Capture Point

As illustrated in Fig. 1, the x, y, and z axes are set to the longitudinal, lateral, and the vertical directions of the robot.

†Authors are with Faculty of Engineering, Department of Mechanical Engineering, University of Kobe, 1-1 Rokkodai, Nada-ku, Kobe, Japan. [gw.kim|kuribayashi]@stu.kobe-u.ac.jp [tazaki|yokokohji]@mech.kobe-u.ac.jp

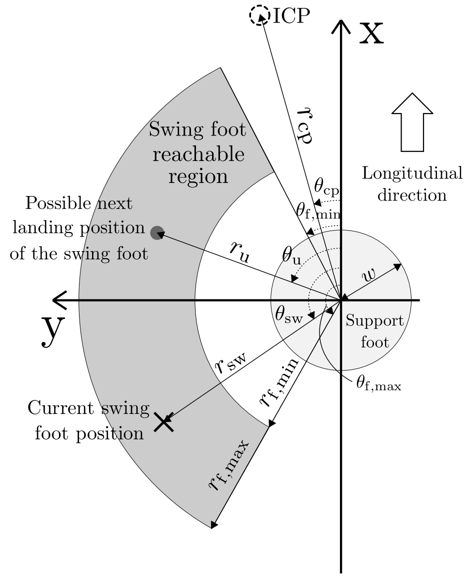


Fig. 1: Top view of the walking model. The center of the light gray circle depicts the current support foot position. The shape of support foot is approximated by a disk whose radius is w. The gray fan-like region depicts the admissible landing position of the swing foot. The values of $r_{f,\max}$, $\theta_{f,\min}$ and $\theta_{f,\max}$ are determined with respect to the kinematic limitations and $r_{f,\min}$ is introduced for preventing self-collision. The dashed circle indicates a possible position of the ICP

It is assumed that the robot makes no rotational movement, and its CoM is kept at a constant height from the level ground (the xy plane). The LIPM describes the motion of the CoM constrained on a horizontal plane. The dynamics of the CoM is derived below:

$$\ddot{\boldsymbol{p}}_{\mathrm{com}} = \omega_0^2(\boldsymbol{p}_{\mathrm{com}} - \boldsymbol{p}_{\mathrm{cop}}) \tag{1}$$

where $\boldsymbol{p}_{\mathrm{com}} = [x_{\mathrm{com}}, y_{\mathrm{com}}]^{\mathsf{T}}$ is the position of the CoM, $\boldsymbol{p}_{\mathrm{cop}} = [x_{\mathrm{cop}}, y_{\mathrm{cop}}]^{\mathsf{T}}$ is the position of the center of pressure (CoP). Moreover, $\omega_0 = \sqrt{g/z_0}$ is the natural frequency of the inverted pendulum, where g is the gravitational acceleration and z_0 is the constant height of a horizontal plane on which the CoM moves.

The ICP is defined as

$$\boldsymbol{p}_{\mathrm{cp}} = \boldsymbol{p}_{\mathrm{com}} + \frac{1}{\omega_0}\dot{\boldsymbol{p}}_{\mathrm{com}} \tag{2}$$

where $\boldsymbol{p}_{\mathrm{cp}} = [x_{\mathrm{cp}}, y_{\mathrm{cp}}]^{\mathsf{T}}$ is the position of ICP. From (1) and (2), the dynamics of ICP is derived as follows

$$\dot{\boldsymbol{p}}_{\mathrm{cp}} = \omega_0(\boldsymbol{p}_{\mathrm{cp}} - \boldsymbol{p}_{\mathrm{cop}}) \tag{3}$$

. Therefore, the trajectory of ICP is derived as shown below by assuming that $\boldsymbol{p}_{\mathrm{cop}}$ is constant during each single support phase.

$$\boldsymbol{p}_{\mathrm{cp}}(t) = [\boldsymbol{p}_{\mathrm{cp}}(0) - \boldsymbol{p}_{\mathrm{cop}}]e^{\omega_0 t} + \boldsymbol{p}_{\mathrm{cop}} \tag{4}$$

.

B. State Space Model

We assume that the robot is pushed with a large disturbance, and make one or more steps to ensure fall-avoidance. To simplify this situation, we introduce a few limitations in the model. The CoP is located at the closest point to the ICP in the support region. When the support foot is exchanged, the CoP is moved to the next support foot instantaneously; continuous-time dynamics during the double support phase is not considered. The swing foot moves to the next landing position at its maximum speed $v_{\max} \in \mathbb{R}$.

The state variable of the system at the k-th step is defined as

$$\boldsymbol{x}_k = \begin{bmatrix} \boldsymbol{d}_{\mathrm{cp},k} \\ \boldsymbol{d}_{\mathrm{sw},k} \end{bmatrix} \in \mathbb{R}^4 \tag{5}$$

where $\boldsymbol{d}_{\mathrm{cp},k}$ and $\boldsymbol{d}_{\mathrm{sw},k}$ are the position of the ICP and the current swing foot with respect to the support foot position, respectively. Here, $\boldsymbol{d} = [r, \theta]^{\mathsf{T}} \in \mathbb{R}^2$ denotes the polar coordinates where r and θ denote the distance from the origin and the direction with respect to the x axis. The control input to the system

$$\boldsymbol{u}_k = \begin{bmatrix} r_{\mathrm{u},k} \\ \theta_{\mathrm{u},k} \end{bmatrix} \in \mathbb{R}^2 \tag{6}$$

is the next landing position of the swing foot.

The step duration is calculated from the current position of the swing foot and the next landing position as:

$$\tau = \frac{dist(\boldsymbol{d}_{\mathrm{sw},k}, \boldsymbol{u}_k)}{v_{\max}} + \Delta t_{\min} \tag{7}$$

where $\Delta t_{\min}$ is the minimum time required for taking off, accelerating and landing the swing foot, and $dist()$ gives the distance of two points in the polar coordinate.

$$dist(\boldsymbol{d}_1, \boldsymbol{d}_2) = \sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos(\theta_1 - \theta_2)} \tag{8}$$

.

For capturability-based analysis, we would like to derive the state of the system after a step specified by $\boldsymbol{u}$. The value of the state after a single step has been taken is expressed as follows in the coordinate frame with respect to the current support foot:

$$\hat{\boldsymbol{x}}_k = \begin{bmatrix} \hat{\boldsymbol{d}}_{\mathrm{cp},k} \\ \hat{\boldsymbol{d}}_{\mathrm{sw},k} \end{bmatrix} \in \mathbb{R}^4 \tag{9}$$

and derived below from (4) and (7)

$$\hat{r}_{\mathrm{cp},k} = (r_{\mathrm{cp},k} - w)e^{\omega_0 \tau} + w \tag{10}$$

$$\hat{\theta}_{\mathrm{cp},k} = \theta_{\mathrm{cp},k} \tag{11}$$

$$\hat{r}_{\mathrm{sw},k} = r_{\mathrm{u},k} \tag{12}$$

$$\hat{\theta}_{\mathrm{sw},k} = \theta_{\mathrm{u},k} \tag{13}$$

.

After the support foot is exchanged, the state variables are reset with respect to the polar coordinates system on the

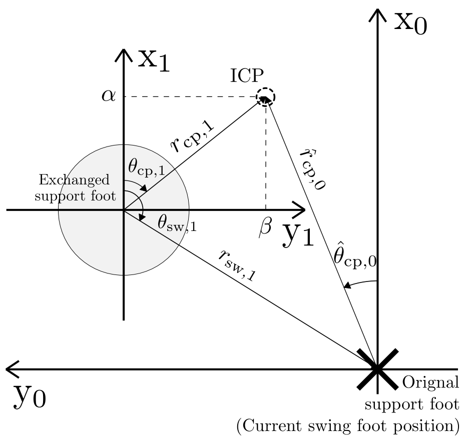


Fig. 2: The illustration of coordinate resetting after exchanging the support foot described by (14) to (17). There are two coordinate systems: one with the subscript 0 and another with the subscript 1. The coordinate system with the subscript 1 is on the ankle of exchanged support foot and its y axis points toward the original support foot. When the support foot is exchanged, the original support foot position (black cross) is regarded as the current swing foot position. Moreover, the state variables are transformed with respect to the exchanged support foot. The dashed circle depicts the position of the ICP at the reset instant.

ankle of exchanged support foot. The state variables after this reset are derived below (see Fig. 2):

$$r_{\mathrm{cp},k+1} = dist(\hat{\boldsymbol{d}}_{\mathrm{cp},k}, \hat{\boldsymbol{d}}_{\mathrm{sw},k}) \tag{14}$$

$$\theta_{\mathrm{cp},k+1} = \begin{cases} \cos^{-1}\left(\frac{\alpha}{dist(\hat{\boldsymbol{d}}_{\mathrm{cp,k}},\hat{\boldsymbol{d}}_{\mathrm{sw},k})}\right) & (\text{if } \beta > 0) \\ 2\pi - \cos^{-1}\left(\frac{\alpha}{dist(\hat{\boldsymbol{d}}_{\mathrm{cp,k}},\hat{\boldsymbol{d}}_{\mathrm{sw},k})}\right) & (\text{otherwise}) \end{cases} \tag{15}$$

$$r_{\mathrm{sw},k+1} = \hat{r}_{\mathrm{sw},k} \tag{16}$$

$$\theta_{\mathrm{sw},k+1} = \pi - \hat{\theta}_{\mathrm{sw},k} \tag{17}$$

where $\alpha = \hat{r}_{\mathrm{cp},k}\cos\hat{\theta}_{\mathrm{cp},k} - r_{\mathrm{u},k}\cos\theta_{\mathrm{u},k}$ and $\beta = -\hat{r}_{\mathrm{cp,k}}\sin\hat{\theta}_{\mathrm{cp},k} + r_{\mathrm{u},k}\sin\theta_{\mathrm{u},k}$. From now on, we use the symbol $f()$ to represent the above step map procedure. Thus, the state at the $(k+1)$-th step is represented as below:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{18}$$

.

C. Constraints

The following inequality constraints are imposed to limit the landing position of the swing foot.

$$\boldsymbol{u}_{\min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{\max} \tag{19}$$

where

$$\boldsymbol{u}_{\min} = \begin{bmatrix} r_{\mathrm{f,min}} \\ \theta_{\mathrm{f,min}} \end{bmatrix}, \boldsymbol{u}_{\max} = \begin{bmatrix} r_{\mathrm{f,max}} \\ \theta_{\mathrm{f,max}} \end{bmatrix} \tag{20}$$

. The set of inputs that satisfy (19) is denoted by U. The valid range depends on the kinematics of a robot under consideration (see Fig. 1 for illustration).

III. OMNI-DIRECTIONAL CAPTURABILITY-BASED ANALYSIS

First, we briefly review the concept of capturability-based analysis in [16]. Let $\boldsymbol{x} \in \mathbb{R}^n$ be the state of a dynamical system and $\mathcal{P}_0 \subset \mathbb{R}^n$ be the set of target states. If there exists at least one sequence of inputs starting from $\boldsymbol{x}$ that reaches $\mathcal{P}_0$ within N steps, $\boldsymbol{x}$ is said to be N-step capturable. Recursively, a state is N-step capturable if there exist inputs that drive the state to a $(N-1)$-step capturable state in one step. The set of all N-step capturable states is called the N-step viable capture basin and represented as $\mathcal{P}_{\mathrm{N}} \subset \mathbb{R}^n$. Furthermore, the set of all feasible foot placements that are able to bring a state into $\mathcal{P}_{\mathrm{N}-1}$ is called the N-step capture region.

1) 0-step viable capture basin: When the distance of the ICP, r_{cp}, is smaller than the foot size w, the robot is able to locate the CoP exactly at the ICP. By definition of ICP, locating the CoP at the ICP brings the pendulum to the upright equilibrium and there is no need to make a step for fall avoidance. Thus, the set of target states, the 0-step viable capture basin $\mathcal{P}_0$, is defined as follows

$$\mathcal{P}_0 = \{\boldsymbol{x} \,|\, r_{\mathrm{cp}} < w\} \tag{21}$$

.

2) N-step viable capture basin: The set $\mathcal{P}_1$ is the set of all states that can be driven to $\mathcal{P}_0$ by some feasible inputs. Given $\mathcal{P}_0$, $\mathcal{P}_1$ is defined as follows:

$$\mathcal{P}_1 = \{\boldsymbol{x} \,|\, \exists \boldsymbol{u} \in U \text{ s.t. } f(\boldsymbol{x},\boldsymbol{u}) \in P_0\} \tag{22}$$

. Recursively, $\mathcal{P}_{\mathrm{N}}$ is derived as

$$\mathcal{P}_{\mathrm{N}} = \{\boldsymbol{x} \,|\, \exists \boldsymbol{u} \in U \text{ s.t. } f(\boldsymbol{x},\boldsymbol{u}) \in P_{\mathrm{N}-1}\} \tag{23}$$

. The N-step capture region is the set of all feasible foot placements that bring a state $\boldsymbol{x}$ into $\mathcal{P}_{N-1}$ in one step. In our capturability-based analysis, $\boldsymbol{u}$ is the next landing position of the swing foot, therefore, we represent the N-step capture region as the set of feasible inputs.

$$\mathcal{U}_{\mathrm{N}}(\boldsymbol{x}) = \{\boldsymbol{u} \,|\, \boldsymbol{u} \in U,\, f(\boldsymbol{x},\boldsymbol{u}) \in \mathcal{P}_{\mathrm{N}-1}\} \tag{24}$$

.

3) ∞-step viable capture basin: When the distance of the ICP is farther than the foot size, the ICP deviates from the CoP exponentially with time, as described by (4). In this case, the robot needs to make a number of steps until it captures the ICP for fall-avoidance. Due to the performance limitation of the actuators, however, there is limit for distance of the ICP that is able to catch with a number of stepping. This limit is described as ∞-step viable capture basin. The N-step viable capture basin will converge to ∞-step viable capture basin as N increases. The ∞-step viable capture basin is denoted by $\mathcal{P}_\infty$.

IV. COMPUTATION OF CAPTURE REGION

A. Algorithm for Capturability-based Analysis

We propose a method to compute approximate N-step viable capture basin and N-step capture region by the discretization of the state space and the input space. First of all, a domain of interest, a subset of the state space in which N-step viable capture basin is computed, is specified as shown below:

$$\mathcal{X} = \{\boldsymbol{x} \mid \boldsymbol{x}_{\min} \leq \boldsymbol{x} \leq \boldsymbol{x}_{\max}\} \tag{25}$$

where

$$\boldsymbol{x}_{\min} = \begin{bmatrix} w \\ 0 \\ r_{\mathrm{f,min}} \\ \theta_{\mathrm{f,min}} \end{bmatrix}, \boldsymbol{x}_{\max} = \begin{bmatrix} r_{\mathrm{cp,max}} \\ 2\pi \\ r_{\mathrm{f,max}} \\ \theta_{\mathrm{f,max}} \end{bmatrix} \tag{26}$$

and the $r_{\mathrm{cp,max}}$ is the upper limit of the distance of the ICP from the support foot. This domain must be set by the designer so that the computed $\mathcal{P}_\infty$ will be included in this domain. A set of grid points X_{grid} is defined over the domain of interest. Let i be a n-dimensional vector of integers ranging from 0 to M, where M denotes the grid resolution. A grid point indexed by i is denoted by $\boldsymbol{x}^i \in X_{\mathrm{grid}}$ and its j-th component is given by

$$x^i{}_j = x_{\min,j} + \frac{x_{\max,j} - x_{\min,j}}{M} i_j \tag{27}$$

. The grid X_{grid} consists of points with all possible indices $i \in \{0, 1, \ldots, M\}^n$. Similarly, another set of grid points U_{grid} is defined over the set of feasible inputs U.

The recursive procedure for computing $\mathcal{P}_{\mathrm{N}}$ and $\mathcal{U}_{\mathrm{N}}(\boldsymbol{x})$ is presented in Algorithm 1. The algorithm takes as the inputs the domain of state space and input space, and the outputs are N-step viable capture basin and capture region. In lines 3-11, the algorithm computes $\mathcal{P}_1$ based on (22) and stores the inputs in $\mathcal{U}^i$. The set $\mathcal{U}^i$ denotes the set of all capture region for $\boldsymbol{x}^i$. In 12-23, the algorithm computes $\mathcal{P}_{\mathrm{N}}$ for $N \geq 2$ based on (23). When $\mathcal{P}_{\mathrm{N}}$ is equal to $\mathcal{P}_{\mathrm{N}-1}$, the capture basin reaches the maximum which is described as ∞-step viable capture basin. In this case, there is no need to compute farther. In line 17, to evaluate if a state is in the $(N-1)$ step viable capture basin, $2^4 = 16$ grid points that surround $\boldsymbol{x}$ are enumerated and defined as the boundary set of $\boldsymbol{x}$, which is denoted by $\mathcal{B}(\boldsymbol{x}) \subset \mathbb{R}^4$.

Algorithm 1 The Algorithm for Viable Capture Basin and Capture Region

```
 1: Input $X_{\mathrm{grid}}, U_{\mathrm{grid}}$
 2: Output $\mathcal{P}_N$, $\mathcal{U}^i$
 3: $N \leftarrow 1, \mathcal{P}_1 \leftarrow \varnothing, \mathcal{U}^i \leftarrow \varnothing$
 4: for each $\boldsymbol{x}^i \in X_{\mathrm{grid}}$ do
 5:     for each $\boldsymbol{u}^j \in U_{\mathrm{grid}}$ do
 6:         $\boldsymbol{x} \leftarrow f(\boldsymbol{x}^i, \boldsymbol{u}^j)$
 7:         if $\boldsymbol{x} \in \mathcal{P}_0$ then
 8:             $\mathcal{P}_1 \leftarrow \mathcal{P}_1 \cup \{\boldsymbol{x}^i\}$, $\mathcal{U}^i \leftarrow \mathcal{U}^i \cup \{\boldsymbol{u}^j, 1\}$
 9:         end if
10:     end for
11: end for
12: while $\mathcal{P}_N \neq \mathcal{P}_{N-1}$ do
13:     $N \leftarrow N + 1$
14:     for each $\boldsymbol{x}^i \in X_{\mathrm{grid}}$ do
15:         for each $\boldsymbol{u}^j \in U_{\mathrm{grid}}$ do
16:             $\boldsymbol{x} \leftarrow f(\boldsymbol{x}^i, \boldsymbol{u}^j)$
17:             $\mathcal{B}(\boldsymbol{x}) \leftarrow$ boundary[$\boldsymbol{x}$]
18:             if $\mathcal{B}(\boldsymbol{x}) \subset \mathcal{P}_{N-1}$ then
19:                 $\mathcal{P}_N \leftarrow \mathcal{P}_N \cup \{\boldsymbol{x}^i\}$, $\mathcal{U}^i \leftarrow \mathcal{U}^i \cup (\boldsymbol{u}^j, N)$
20:             end if
21:         end for
22:     end for
23: end while
```

V. FALL AVOIDANCE CONTROLLER

Fig. 3 shows a block diagram of the proposed fall-avoidance controller that can be used in combination with any conventional foot-step planner that produces desired foot steps. The controller takes the current CoM position, velocity, current swing foot position, and nominal reference landing position provided by the foot-step planner. The controller provides the number of required steps to bring the robot to a target state in $\mathcal{P}_0$, step duration, and the modified landing position of the swing foot.

1) Precomputed Capture Region Database: The N-step capture regions are computed with Algorithm 1 and stored into a database. Since N-step capture regions depend only on the physical parameters of a humanoid robot and not on its state, there is no need to compute the capture region in real time. One notable characteristic of Algorithm 1 is that it is highly parallelizable. In fact, the algorithm was implemented on GPGPU as described in Section VI.

2) Capture region calculator: The capture region calculator reads the capture region from stored data. It searches the nearest grid $\boldsymbol{x}^i$ from current state $\boldsymbol{x}$

$$\min_i ||\boldsymbol{x} - \boldsymbol{x}^i||_2^2 \tag{28}$$

and retrieves the capture region $\mathcal{U}^i$ associated with it. The capture region $\mathcal{U}^i$ consists of feasible landing positions of the swing foot and the number of steps N which is needed to reach the target states.

3) Desired landing position calculator: From capture region calculator, the desired landing position calculator gets the candidates of the landing position. There are two criteria for deciding a single point in the capture region. First, N with smaller value have great priority. Taking small value of N makes the robot balanced in shorter time. Second, the nearest point from the reference landing position takes precedence in the capture region with the same N. It minimizes the influence of the modification of landing position.

VI. NUMERICAL RESULTS

A. Capturability Computation Results

In the following case studies, capture regions are computed with parameters listed in Table I. The values of the parameters are set assuming a small-sized humanoid robot such as NAO [17]. Algorithm 1 was executed on a computer with GTX 960 graphics board, which has 1024 CUDA cores and 2 GB VRAM, for general-purpose computing on graphics processing units (GPGPU). The entire computation time was

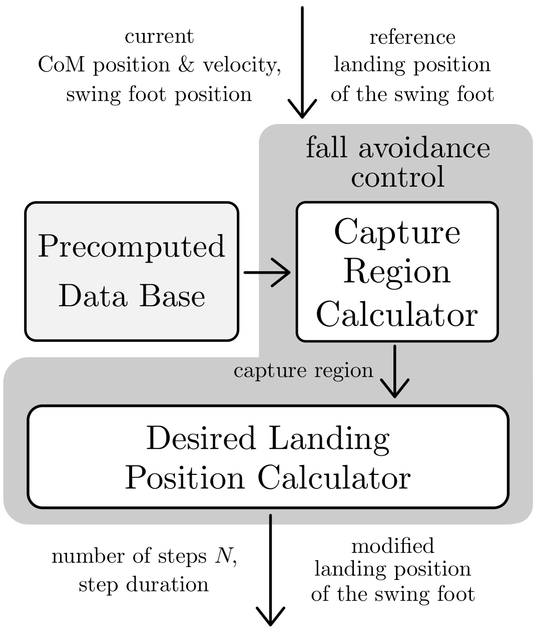


Fig. 3: Structure of the fall-avoidance controller. Arrows represent data flow. The capture region calculator reads the precomputed data and provides the capture region that depends on the current state. The desired landing position calculator decides the next landing position of the swing foot among the capture region and provides the number of steps that remains until the state reaches $\mathcal{P}_0$.

TABLE I: Parameters used for capturability computation

Parameter	Symbol	Value	Units
Foot size	w	0.04	[m]
Height of CoM	z_0	0.3	[m]
Gravity	g	9.81	[m/s^2]
Swing foot velocity	v	1.0	[m/s]
Grid resolution	M	20	
Min. step duration	$\Delta t_{\min}$	0.1	[s]
Max. ICP	$r_{\mathrm{cp,max}}$	0.2	[m]
Max. step length	$r_{\mathrm{f,max}}$	0.22	[m]
Min. step length	$r_{\mathrm{f,min}}$	0.09	[m]
Max. step angle	$\theta_{\mathrm{f,max}}$	160	[deg]
Min. step angle	$\theta_{\mathrm{f,min}}$	20	[deg]

approximately 20 minutes, and the output data size was approximately 140 MB.

The computed capture regions for three different states are visualized in Fig. 4. When the swing foot position is close to the ICP (Case 1), there is a wide capture region, but when the swing foot position is far from the ICP (Case 2), the capture region becomes much smaller. In Case 3, the ICP is in the opposite side of the swing foot. In this Case, the ICP cannot be captured in a single step due to the kinematic limitations of swing foot placement. As a result, a small region is calculated as 2-step capture region. In Fig. 4a, 3-step capture region is observed between 1-step and 2-step capture regions. This seemingly unnatural result is caused by lack of consideration about continuous-time dynamics during the double support phase.

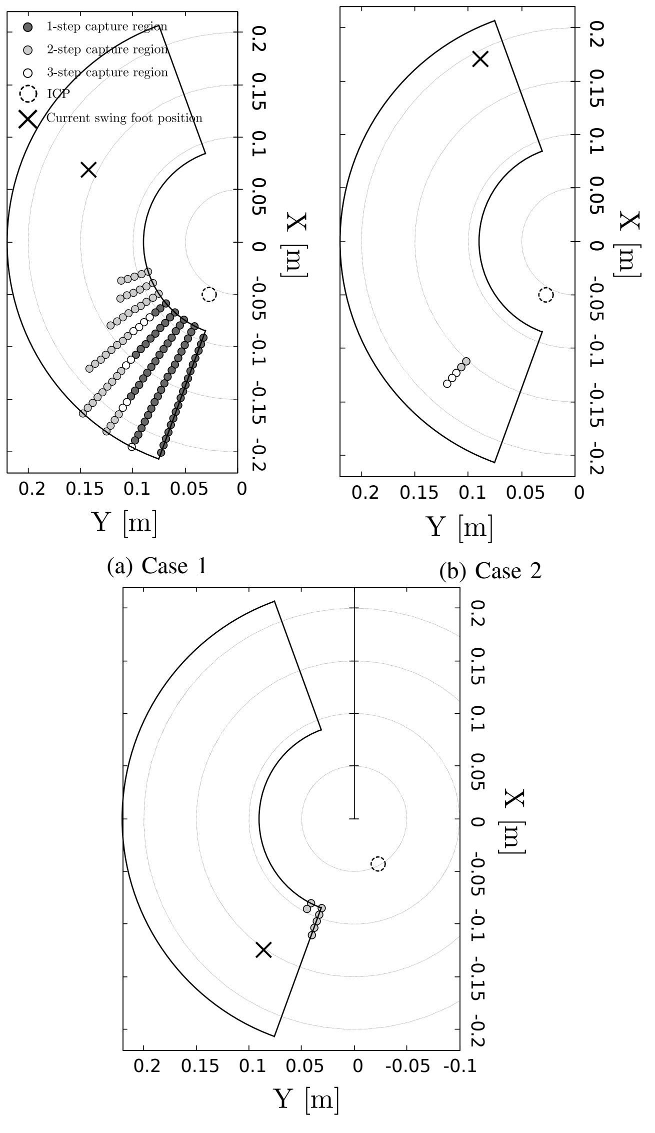


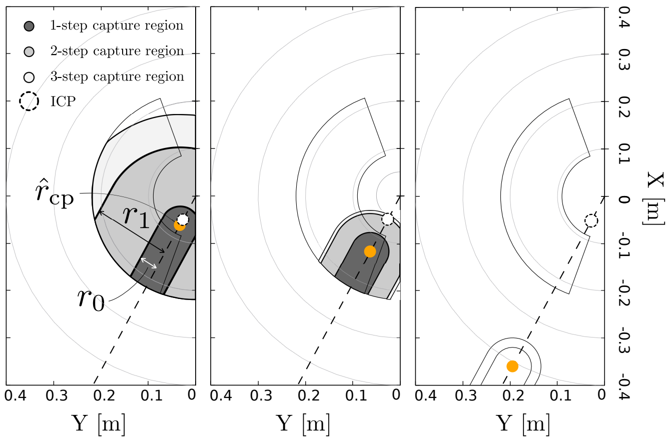
Fig. 4: The N-step capture regions computed by the proposed method. The center of standing foot is located at the origin of graphs. The fan-like region depicts the admissible landing position of the swing foot. The black cross point is the current swing foot position and the dashed circle indicates a position of the ICP.

B. Comparison with Conventional Capturability Analysis

Capture regions computed by the proposed method and the conventional method [16] are compared. The conventional method determines the capture regions in the following steps. First, the set of possible ICP locations at the next swing-foot touch-down is determined. Next, a series of nested regions around this set is constructed based on the following formula:

$$r_{\mathrm{N}} = (l_{\max} - w + r_{\mathrm{N-1}})e^{-\omega_0 \tau_{\mathrm{s}}} + w \qquad (29)$$

where r_{N} is the maximum distance of the N-step capture point and $r_0 = w$. Here, step length $l_{\max}$ is the maximum distance between subsequent ankle location in ICP direction.



(a) $\tau_{\mathrm{s}} = 0.1$[s] (b) $\tau_{\mathrm{s}} = 0.32$[s] (c) $\tau_{\mathrm{s}} = 0.52$[s]

Fig. 5: The N-step capture regions based on conventional capturability analysis. The orange points are the ICP location at the minimum step duration τ_{s}. The dashed circle indicates a position of the ICP

The conventional method treats the step duration τ_s as a constant. To compare it with our variable step duration setting, three cases were considered. From (7), the minimum value of τ_{s} is 0.1[s] where the stride length is 0, $\tau_{\mathrm{s}} = 0.32$[s] where the stride length is $r_{\mathrm{f,max}}$, and the maximum value of τ_{s} is 0.52[s] where the stride length has the maximum value. The value of $l_{\max}$ is simple set to $r_{\mathrm{f,max}} = 0.22$[m]. The N-step capture regions calculated by using the conventional method are shown in Fig. 5. In the conventional method, the location of the CoP is arbitrary inside the support region, while in our setting, it is constrained to the nearest point to the ICP. This limitation of the model causes underestimation of the capturability of the robot to recover compared to the conventional method. We admit that this is one of the limitations of our method, but for fair comparison, at this time, we imposed this constraint to the conventional method.

There are two notable points compared to the conventional method. First, the capture region is variable with respect to the current position of the swing foot. In the conventional method, when τ_{s} is 0.1[s] (Fig. 5a), the capture regions are nested near to the ICP, while when τ_{s} is 0.32[s] (Fig. 5b), the capture regions are far from the ICP. Although fairly large capture regions are observed in Fig. 5a, since the conventional method lacks the information of the current swing foot position, there is no assurance that the swing foot can reach any points in these regions within τ_{s}. To assure that the swing foot reach any points in the admissible landing position, the duration should be bigger than 0.52[s]. However, when τ_{s} is 0.52[s] (Fig. 5c), the possible ICP locations at τ_{s} are out of swing foot reachable region, thus, there are no capture regions in the admissible landing position. The proposed method is able to deal with this issue as explained below. In Fig. 4a and Fig. 4b, Case 2 has the same ICP with Case 1, but the current swing foot position is far from the ICP. In Case 2, it takes more time to reach the ICP, and as a result, the computed capture regions are much smaller than that of Case 1. Second, it can find the capture regions in the presence of lateral disturbance. When the robot executes lateral stepping for fall-avoidance, the step length is no more constant. For this reason, the conventional method cannot find the capture regions in lateral way. The proposed method, however, is able to find the capture region when the ICP is in the negative y side of the plane as shown in Fig. 4c. From this capture region in Case 3, we can determine the next foot step position that is guaranteed to balance in two steps.

C. Simulation Results

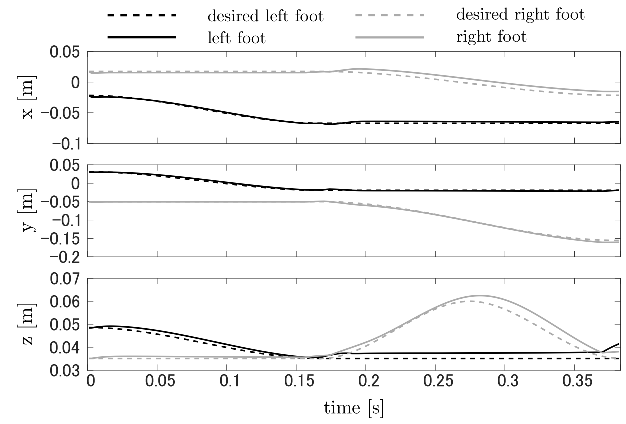
To evaluate the effectiveness of the proposed fall avoidance controller, a case which requires two steps to avoid falling down is tested on a robot simulator Choreonoid [18]. The result of simulation is shown in Fig. 6. A humanoid robot (24 DOFs, 2.498[kg] weight, 573[mm] height) that is standing at the right foot (Initial support foot in Fig. 6b) is pushed into the opposite side of the swing foot at time 0[s]. Impulsive disturbance with the magnitude $[-650, -350, 0]^{\mathsf{T}}$[N] and the duration 1[ms] is applied to the CoM, which gives $[-0.28, -0.16, 0]^{\mathsf{T}}$[m/s] velocity to the CoM. According to our capturability analysis, the robot needs to make two steps to avoid falling in this situation.

For simplification, the nominal reference landing position is determined at next to the support foot. The trajectories of swing foot are obtained by third order polynomials between the current position and the next landing position. And then, desired joint angles are calculated by using inverse kinematics library in Choreonoid. The robot moves the left foot from the initial swing foot position to Step 1 and exchanges the support foot at time 0.16[s]. After the support foot is exchanged, the robot moves its right foot to Step 2 and is able to catch the ICP at time 0.37[s]. Thanks to the fact that capture regions are precomputed, the computation time of the online fall-avoidance controller at every control cycle was approximately 1 ms. As a result, the robot is able to recover balance from the disturbance using the proposed fall avoidance controller.

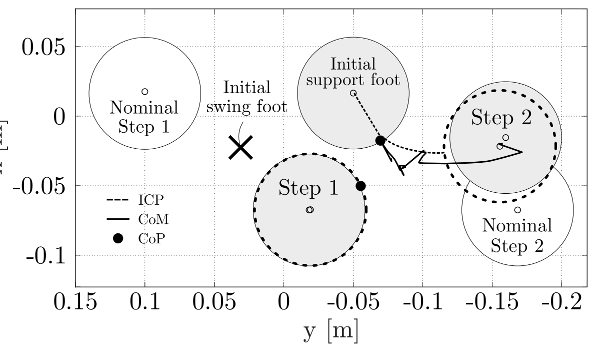
In this experiment, the parameters were set assuming a small-sized humanoid robot that has relatively large footprint and thick legs compared to human in the same scale. Considering self-collision, these features result in narrowing the foot reachable region and make it hard to capture the ICP when its direction is toward the opposite side of the swing foot. This is one of the reasons for which a very small capture region was calculated in Case 3, and that quite fast swing foot velocity 1.0[m/s] is required. As a result, it gets the step durations, 0.16[s] and 0.21[s], which are quite difficult to be obtained outside of simulations. Resolving the limitations mentioned in the previous subsection is also expected to ease this problem.

VII. Conclusion

This paper proposed capturability-based analysis that considers both the relation between stride length and step



(a) Foot trajectory



(b) Top view of ICP and CoM trajectory

Fig. 6: Simulation result: an impulsive disturbance is applied when the robot is standing still on its right foot. (a) The left and right foot trajectory. The support foot is exchanged twice at time 0.16[s] and 0.37[s], respectively. (b) The white circles are the nominal reference foot placement, the dashed circles are modified foot placement, the gray circles are the actual foot placement, and the black cross point is the initial swing foot position.

duration, and the kinematic limitation of foot placement. It was shown that the proposed method is able to compute more realistic capture regions compared to conventional methods. Future work includes extending the present framework to handle double support phase, and testing our method on real humanoid robots.

ACKNOWLEDGMENT

This research was supported by NSK Foundation for Advancement of Mechatronics.

REFERENCES

[1] B. Stephens : "Humanoid Push Recovery", *IEEE-RAS International Conference on Humanoid Robots*, pp.589–595, 2007.

[2] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa : "The 3D Linear Inverted Pendulum Mode: A Simple Modeling for a Biped Walking Pattern Generation", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol.1, pp.239–246, 2001.

[3] J. Pratt, J. Carff, S. Drakunov, and A. Goswami: "Capture Point: a Step Toward Humanoid Push Recovery" *IEEE-RAS International Conference on Humanoid*, pp.200–207, 2006.

[4] T. Takenaka, T. Matsumoto, and T. Yoshiike: "Real Time Motion Generation and Control for Biped Robot - 1st Report: Walking Gait Pattern Generation", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.1084–1091, 2009.

[5] T. Koolen, S. Bertrand, G. Thomas, T. De Boer, T. Wu, J. Smith, J. Englsberger, and J. Pratt: "Design of a Momentum-Based Control Framework and Application to the Humanoid Robot Atlas", *International Journal of Humanoid Robotics*, Vol. 13, No.1, pp.165007-1 – 165007-34, 2016.

[6] M. A. Hopkins, D. W. Hong, and A. Leonessa: "Compliant Locomotion Using Whole-Body Control and Divergent Component of Motion Tracking", *IEEE International Conference on Robotics and Automation*, pp.5726–5733, 2015.

[7] A. Herdt, H. Diedam, P.-B. Wieber, D. Dimitrov, K. Mombaur, and M. Diehl: "Online Walking Motion Generation with Automatic Foot Step Placement", *Advanced Robotics*, 24 (5–6), pp.719–737, 2010.

[8] R. J. Griffin and A. Leonessa: "Model predictive control for dynamic footstep adjustment using the divergent component of motion", *IEEE International Conference on Robotics and Automation*, pp. 1763–1768, 2016.

[9] M. Khadiv, A. Herzog, S. A. A. Moosavian, and L. Righetti: "Step Timing Adjustment : A Step Toward Generating Robust Gaits", *IEEE-RAS International Conference on Humanoid Robots*, pp.35–42, 2016.

[10] T. Sugihara and T. Yamamoto: "Foot-Guided Agile Control of a Biped Robot through ZMP Manipulation", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.4546–4551, 2017.

[11] K. Yamamoto: "Control Strategy Switching for Humanoid Robots Based on Maximal Output Admissible Set", *Robotics and Autonomous Systems*, Vol.81, No.2, pp.17–32 , 2016.

[12] T. Koolen, M. Posa, and R. Tedrake: "Balance Control Using Center of Mass Height Variation: Limitations Imposed by Unilateral Contact", *IEEE-RAS International Conference on Humanoid Robots*, pp.8–15, 2016.

[13] T. Sugihara and Y. Nakamura: "Boundary Condition Relaxation Method for Stepwise Pedipulation Planning of Biped Robots", *IEEE Transations on Robotics*, Vol.25, No.3, pp.658–669, 2009.

[14] R. J. Griffin, G. Wiedebach, S. Bertrand, A. Leonessa, and J. Pratt: "Walking Stabilization Using Step Timing and Location Adjustment on the Humanoid Robot, Atlas", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.667–673, 2017.

[15] K. Goto, Y. Tazaki, and T. Suzuki: "Bipedal Locomotion Control Based on Simultaneous Trajectory and Foot Step Planning" *Journal of Robotics and Mechatronics*, Vol.28, No.4 pp.533–542, 2016.

[16] T. Koolen, T. de Boer, J. Rebula, A. Goswami, and J. Pratt: "Capturability-Based Analysis and Control of Legged Locomotion, Part 1: Theory and Application to Three Simple Gait Models", *The International Journal of Robotics Research*, Vol.31, pp.1094–1113, 2012.

[17] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier : "Mechatronic Design of NAO Humanoid", *IEEE International Conference on Robotics and Automation*, pp.767–774, 2009.

[18] S. Nakaoka: "Choreonoid: Extensible Virtual Robot Environment Built on an Integrated GUI Framework", *IEEE/SICE International Symposium on System Integration*, pp.79–85, 2012.