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

Goal-Oriented Simulation-Based Motion Interpolator for Complex Contact Transition: Experiments on Knee-Contact Behavior

Shintaro Noda, Yohei Kakiuchi, Hiroki Takeda, Kei Okada and Masayuki Inaba

***Abstract*— It is in process to build robust robotics system enabling whole-body multi-contact motion. In this paper, we have experiments on knee-contact motions to preliminary investigate motion planning algorithm to generate whole-body multi-contact behavior. Our motion interpolator is goal-oriented in that the interpolator does not specify detailed contact constraints such as fixed contact point on link, friction cone constraints and timing of contact switching. The goal-oriented feature enables to generate complex contact transition including sliding, rotating and dynamic contact transition. The interpolator generates whole-body trajectory to achieve goal state considering physical feasibility such as whole-body dynamics, collision, and joint torque limitations by using dynamics simulator. Further, the generated knee-contact motions are achieved by actual humanoid robot RHP4B to check difference between simulated motion and actual result.**

I. Background and Motivation

RHP4B robot [1] demonstrates standing up after slow falling to shows that almost all of body surface are possible to contact with environment. Among the research, we need motion planning algorithm to deal with whole-body multi-contact behavior. In this paper, we have experiments on knee-contact motion to preliminarily investigate whole-body multi-contact behavior. We have three reasons to choose knee-contact motion; First reason is that knee-contact motion is efficient as relay point between foot contact state such as standing posture and whole-body contact state (Fig. 1). Second reason is that the motion is safe and easy due to large support region and low CoG (Center of Gravity) of robot. Low CoG is efficient to falling experiments. For example, although there are a lot of researches on falling motion of robot [2][3][4], there are few experiments with actual and especially large-size robot because robot breaks after falling. However, low CoG motion is relatively safe to fall, and the safeness is adjustable by changing the CoG height. Third reason is that knee-contact motion includes difficult feature of whole-body multi-contact motion. Knee link is generally not simple plane which is frequently used as foot contact model of walking [5] but spherical shape. Although the shape enables various contacts such as rotating and sliding, motions including such contacts are difficult to be planned but are necessary for whole-body multi-contact behavior. To deal with such complex contacts, both of motion planner and balancing controller become complex. In previous research on balancing control, ZMP-based contact force control [6][7] and whole-body torque-based control

The authors are with Graduate School of Information Science and Technology, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan (s-noda at jsk.t.u-tokyo.ac.jp).

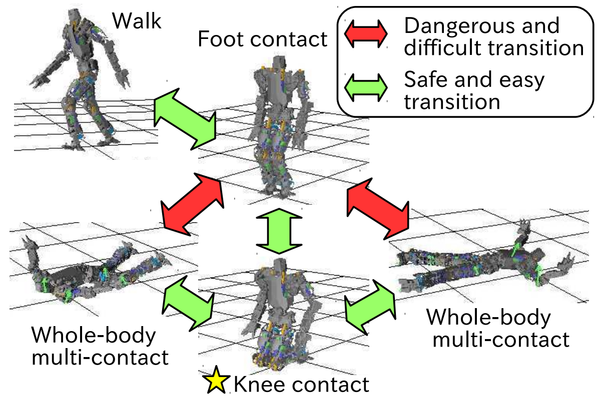


Fig. 1. Knee contact motion plays key role in whole-body multi-contact behavior as preliminarily experiments, relay posture to transit between foot contact, and to safely experiment on falling motions.

[8] are proposed. Although we focus on motion planner in this paper, we assume that such balancing controller will be applicable to our planner. Previous researches on motion planner use different types of algorithms including configuration search after contact search [9][10], complementarity condition to consider contact or not contact at the same time to search configuration space [11], and simulation-based evolutionary search [12]. In this paper, we use simulation-based approach in the following three reasons; First, we assume that sampling-based search algorithm is better for whole-body multi-contact behavior because motion planning considering a lot of contact points is non-convex optimization problem involving a lot of local solutions. Second, simulation-based algorithm is possible to consider all contact states if the states can be simulated. The merit is beneficial to complex contact motion. Third, it is not necessary to specify detailed contact condition. For example, our simulation-based algorithm does not specify which part of link will have contact, when the contact will be detached and what kind of contact condition (e.g. sliding or rotating) will be. Instead, the algorithm considers goal condition and physical feasibility such as whole-body dynamics and collision. Such goal-oriented feature enables to generate complex contact behavior. In the following sections, we show three experiments on knee-contact motions: sitting down from foot contact to knee contact, standing up from knee contact to foot contact and rotating with knee contact. The three problems are solved in simulation world and achieved with actual RHP4B robot to confirm computational time of planning and reproductivity in real world. The purpose of this paper is to investigate whole-body multi-contact behavior through experiments on knee-contact motions.

II. Method to generate knee-contact motions

An evolutionary search algorithm with dynamics simulation is described in this section. The algorithm considers whole-body dynamics and whole-body contacts with ground satisfying joint torque and contact wrench limitations.

Algorithm 1 Pipeline of motion search using simulator

Require: t_m : maximum simulation time, Δt : time step.
Variable: t : current time in simulation world, $\boldsymbol{q}_d$: target joint angle vector, $\mathcal{G}$: array of genes, $\mathcal{O}$: array of objective value of genes.

Procedure: evolutionary_search_motion() # Main loop similar to [13] implemented in NLopt [14]

$\mathcal{G} \Leftarrow$ uniform_random_genes()
while **not** is_search_converged() **do**
  for all $\boldsymbol{g} \in \mathcal{G}$, $\boldsymbol{o} \in \mathcal{O}$ **do**
    $\boldsymbol{o} \Leftarrow$ eval_gene($\boldsymbol{g}$) # Parallel computation here
  end for
  $\mathcal{G} \Leftarrow$ sort_by_objective($\mathcal{G}$, $\mathcal{O}$)
  $\mathcal{G} \Leftarrow$ update_top30%_by_amoeba_method($\mathcal{G}$)
  $\mathcal{G} \Leftarrow$ mutate_last70%_by_Gauss_distribution($\mathcal{G}$)
end while
return $\mathcal{G}$

Procedure: eval_gene($\boldsymbol{g}$)

initialize_simulator() # Initialize robot state in simulator
while $t \le t_m$ **do**
  $\boldsymbol{q}_d \Leftarrow$ calc_Lagrange_polynomial($\boldsymbol{g}$, t, t_m) # Eq. (1)
  step_simulation($\boldsymbol{q}_d$)
  if is_motion_invalid() **then**
    break # Error such as self collision as Eq. (5)
  end if
  $1/S + P \Leftarrow$ update_score_and_penalty($1/S + P$)
  $t \Leftarrow t + \Delta t$
end while
return $1/S + P$ # Score and penalty (Eq. (3), (4))

Algorithm 2 Simplified code of simulator

Require: $\boldsymbol{q}$: joint angle vector, Δt : time step, $K_{p,d}$: PD gain matrix of position control, $k^v_{p,d}$: PD gain to calculate vertical force from ground insertion, $k^h_{p,d}$: PD gain to calculate friction force from sliding distance, $\boldsymbol{n}$: Ground normal vector, $\boldsymbol{x}_0$: Ground position, μ : Ground friction coefficient.
Variable: $\mathcal{X}$: array of positions of all vertices to collide, $\mathcal{V}$: array of velocities of all vertices to collide, $\mathcal{F}$: array of contact forces on all vertices to collide, $\mathcal{X}'$: previous $\mathcal{X}$, $\boldsymbol{\tau}$: target joint torque vector.

Procedure: step_simulation($\boldsymbol{q}_d$) # Simulate in one time step

$\boldsymbol{\tau} \Leftarrow K_p(\boldsymbol{q}_d - \boldsymbol{q}) + K_d\dot{\boldsymbol{q}}$ # PD position control
forward_dynamics($\boldsymbol{\tau}$, $\mathcal{F}$, Δt) # Featherstone's method [15] to calculate acceleration from joint torque and Euler method to integrate acceleration and velocity
[$\mathcal{X}$, $\mathcal{V}$] $\Leftarrow$ forward_kinematics() # All positions and velocities of all vertices to collide
$\mathcal{F} \Leftarrow$ calc_contact_force($\mathcal{X}$, $\mathcal{V}$)

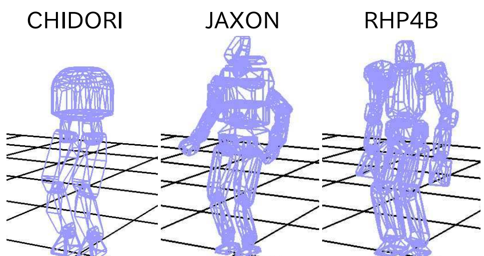
Procedure: calc_contact_force($\mathcal{X}$, $\mathcal{V}$)

for all $\boldsymbol{x} \in \mathcal{X}$, $\boldsymbol{v} \in \mathcal{V}$, $\boldsymbol{x}' \in \mathcal{X}'$, $\boldsymbol{f} \in \mathcal{F}$ **do**
  if $(\boldsymbol{x} - \boldsymbol{x}_0)^T\boldsymbol{n} < 0$ **then**
    $f_z \Leftarrow k^v_p(\boldsymbol{x} - \boldsymbol{x}_0)^T\boldsymbol{n} + k^v_d\boldsymbol{v}^T\boldsymbol{n}$ # Vertical force
    if $\boldsymbol{x}' = NULL$ **or** $f_z \le 0$ **then**
      $\boldsymbol{f} \Leftarrow$ max(0, f_z) $\boldsymbol{n}$ # No friction force
    else
      $\boldsymbol{h} \Leftarrow \boldsymbol{x} - \boldsymbol{x}'$ # Friction direction
      $f_x \Leftarrow k^h_p\boldsymbol{h} + k^h_d\boldsymbol{v}^T\boldsymbol{h}/||\boldsymbol{h}||$
      if $||f_x|| > ||\mu f_z||$ **then**
        $f_x \Leftarrow f_x \times ||\mu f_z||/||f_x||$ # Sliding
      else
        $\boldsymbol{x} \Leftarrow \boldsymbol{x}'$ # No sliding and keep friction center
      end if
      $\boldsymbol{f} \Leftarrow f_z\boldsymbol{n} + f_x\boldsymbol{h}/||\boldsymbol{h}||$
    end if
  else
    $\boldsymbol{f} \Leftarrow \boldsymbol{0}$, $\boldsymbol{x} \Leftarrow NULL$ # No collision
  end if
end for
$\mathcal{X}' \Leftarrow \mathcal{X}$ # Backup previous collide positions
return $\mathcal{F}$

A. Pipeline of motion search: settings of dynamics simulator and evolutionary search algorithm

Pipeline of motion search using simulator is shown in Algorithm 1. Main loop of the search is defined as evolutionary_search_motion(). We use an evolutionary search algorithm which involves mutation and amoeba method [13]. We preliminarily checked search capabilities of several non-linear optimization algorithms implemented in NLopt [14], and the evolutionary algorithm was the best. We reimplement the algorithm by C++ language to enable parallel computation in evaluation of all genes by using shared memory of Ubuntu OS. The search checks if simulation time is less than maximum simulation time t_m, updates score and penalty function mentioned later ($1/S + P$, Eq. (3), (4)) and stops calculation if invalid results appear such as self collision.

Pseudo code of our simple and fast simulator is shown as Algorithm 2. We use Featherstone's approach [15] to solve forward dynamics (forward_dynamics()), and linear compliant contact model [16] to calculate contact force (calc_contact_force()). To simulate friction force, Simbody [17] uses a function of horizontal velocity. In contrast, we use both of horizontal velocity and distance from the first collided point to express inner force in contact surface. The pseudo code of contact force calculation are simplified to explain the experiments in this paper. For example, the pseudo code consider only one plane as contact environment because we demonstrate motions in a flat ground. PD gains and friction coefficient of contact force calculation are common to all vertices because all body parts of RHP4B are equally covered by metal frame. However, the actual simulator implementation can consider collision between convex bodies and different settings of contact model for each body part. To measure simulation speed, we simulate falling down motion of RHP4B [1], JAXON [18], CHIDORI [19] robots by using Intel(R) Core(TM) i7-4770S CPU 3.10GHz. Result is shown in Fig. 2. Blue lines connect all vertices to consider collision of robots. RHP4B robot has 1322 vertices to collide and 32 joints. The simulation speed is 21.7 kHz. Because we use 1 ms as simulation time step in the following experiments, the simulation speed is 21.7 times faster than real world. The time step is limited to less than the fastest controller of actual robot. In our case, We use 1 ms to simulate position controller.



12 DoF	33 DoF	32 DoF
642 verts	1799 verts	1322 verts
51.4 kHz	19.0 kHz	21.7 kHz

Fig. 2. Simulation speed, joint DoF and all vertices to consider collision

B. Search space and objective function

We use polynomial function to search whole-body configuration trajectory and define cost and penalty function to consider collision, joint torque and contact wrench limitations. Symbols to explain the definition are listed in TABLE. I - III in advance. We use Lagrange polynomial to express joint angle trajectory and use control points of the polynomial as search space. The other choices of polynomial function include Bezier curve [20] and basic spline [21][22]. We use Lagrange polynomial because it is easy to implement. However, there is a risk of divergence to connect large number of control points (Runge's phenomenon). In this paper, we use small number of control points (M=3) and generated motion does not diverge. It is our future work to investigate other polynomial for more complex motion. Definition of joint angle trajectory $\boldsymbol{q}$ is as follows:

$$\boldsymbol{q} = \boldsymbol{L}(\boldsymbol{Q};\boldsymbol{x},T(t)) \tag{1}$$

$$s.t. \left\{ \begin{array}{lcl} \boldsymbol{Q} & = & [\boldsymbol{q}_0 \cdots \boldsymbol{q}_{M-1}] = [\boldsymbol{q}^0 \cdots \boldsymbol{q}^{N-1}]^T \\ & = & \begin{pmatrix} q_0^0 & \cdots & q_{M-1}^0 \\ \vdots & \ddots & \vdots \\ q_0^{N-1} & \cdots & q_{M-1}^{N-1} \end{pmatrix} \\ \boldsymbol{x} & = & [x_0, \cdots, x_{M-1}]^T \end{array} \right.$$

Definitions of function L and T are as follows:

$$\boldsymbol{L}(\boldsymbol{Q};\boldsymbol{x},t) = [L(\boldsymbol{q}^0;\boldsymbol{x},t) \cdots L(\boldsymbol{q}^{N-1};\boldsymbol{x},t)]^T \tag{2}$$

$$L(\boldsymbol{q}^k;\boldsymbol{x},t) = \sum_{i\in[0,M)} q_i^k \prod_{j\in[0,M),i\neq j} \frac{t-x_j}{x_i-x_j}$$

$$T(t) = \left\{ \begin{array}{ll} 0 & t \leq 0 \\ 1 & t \geq t_m \\ t/t_m - sin(2\pi t/t_m)/2\pi & else \end{array} \right.$$

TABLE I

SYMBOLS OF SEARCH

	DIM	Description
N	1	Total number of joints
M	1	Total number of Lagrange polynomial control points
t_m	1	Interpolation duration of joint angle trajectory
t	1	Interpolation time $t \in [0, t_m]$
Δt	1	Simulation time step $= 1$ [ms]
$1/S$	1	Inverse of score of motion to minimize
P	1	Penalty of motion to minimize
$\boldsymbol{q}$	N	Joint angle vector depending on t
$\boldsymbol{\tau}$	N	Joint torque vector depending on t
$\boldsymbol{w}$	N	Joint velocity vector depending on t
$\boldsymbol{p}$	3	Position vector of root link depending on t
$\boldsymbol{r}$	3	Rotation vector in Rodrigues' form depending on t
$\boldsymbol{f}_{l,r}$	6	Both foot contact wrench. Function of time t

TABLE II

ALL WEIGHT CONSTANT PARAMETERS OF OBJECTIVE OF SEARCH

	Constant Value	Description
w_t	10	Score weight of motion duration
w_e	1	Score weight of joint torque and velocity
W_t	10^6	Penalty weight of time when error occurred
W_r	10^6	Penalty weight of root distance from target
W_f	10^3	Penalty weight of foot contact wrench
W_e	10^6	Penalty weight of joint torque and velocity
W_c	10^6	Penalty weight of collision

TABLE III

ALL CONSTRAINT CONSTANT PARAMETERS OF SEARCH

	Constant Value	Description
$\boldsymbol{e}$	$[1, 1, 1]^T$	3d vector, all elements are 1
$\boldsymbol{p}^-$	$0.1\boldsymbol{e}$ [m]	Soft max root distance from target
$\boldsymbol{p}^+$	$1\boldsymbol{e}$ [m]	Hard max root distance from target
$\boldsymbol{r}^-$	$0.2\boldsymbol{e}$ [rad]	Soft max root slope from target
$\boldsymbol{r}^+$	$0.5\pi\boldsymbol{e}$ [rad]	Hard max root slope from target
$\boldsymbol{f}_{l,r}^-$	$[2\boldsymbol{e}^T$[kN], $50\boldsymbol{e}^T$[Nm]$]^T$	Soft max foot contact wrench
$\boldsymbol{f}_{l,r}^+$	$[4\boldsymbol{e}^T$[kN], $100\boldsymbol{e}^T$[Nm]$]^T$	Hard max foot contact wrench
$\boldsymbol{\tau}^-$	$300\boldsymbol{e}$ [Nm]	Soft max joint torque
$\boldsymbol{\tau}^+$	$600\boldsymbol{e}$ [Nm]	Hard max joint torque
$\boldsymbol{w}^-$	$\pi\boldsymbol{e}$ [rad/s]	Soft max joint velocity
$\boldsymbol{w}^+$	$2\pi\boldsymbol{e}$ [rad/s]	Hard max joint velocity
P_c^+	100	Hard max collision count

The function $\boldsymbol{L}$ expresses Lagrange polynomial which connects each given joint angles $\boldsymbol{q}_j, j \in [0, M)$ at a given time $t = x_j$ in N-dimensional space. The time t is larger than 0 and smaller than interpolation duration t_m. The function T increases monotonically from $T(0) = 0$ to $T(t_m) = 1$. The velocity and acceleration are zero if $t = 0$ or $t = t_m$. Search space includes $\boldsymbol{q}_0 \cdots \boldsymbol{q}_{M-1}$ and t_m. Definition of score function S is as follows:

$$1/S = w_t t_m + w_e \sqrt{\sum \frac{||\boldsymbol{\tau}||^2}{N} \Delta t} + w_e \sqrt{\sum \frac{||\boldsymbol{w}||^2}{N} \Delta t} \tag{3}$$

First term means fast motion is better (interpolation duration t_m is small). Second term means small joint torque is better. $\boldsymbol{\tau}$ is joint torque vector of all joints and N is total number of joints. Third term means small joint velocity is better. $\boldsymbol{w}$ is joint velocity vector of all joints. w_t, w_e are constant values to weight scores. Definition of penalty function is as follows:

$$P = W_t P_t + W_r P_r + W_f P_f + W_e P_e + W_c P_c \tag{4}$$

$$
s.t. \begin{cases} P_t &= t_m - t \\ P_r &= B(\boldsymbol{p}, \boldsymbol{p}^-, \boldsymbol{p}^+) + B(\boldsymbol{r}, \boldsymbol{r}^-, \boldsymbol{r}^+) \\ P_f &= B(\boldsymbol{f}_l, \boldsymbol{f}_l^-, \boldsymbol{f}_l^+) + B(\boldsymbol{f}_r, \boldsymbol{f}_r^-, \boldsymbol{f}_r^+) \\ P_e &= B(\boldsymbol{\tau}, \boldsymbol{\tau}^-, \boldsymbol{\tau}^+) + B(\boldsymbol{w}, \boldsymbol{w}^-, \boldsymbol{w}^+) \\ P_c &= Total\ Number\ of\ Collision \end{cases}
$$

P_t imposes penalty on simulation time. t is simulation time which is equal to t_m if simulated motion is valid and is_motion_invalid() in Algorithm 2 always returns false. If simulated motion is invalid, t becomes smaller than t_m and P_t becomes positive value (penalty). The conditions of invalid simulation are as follows:

$$
\begin{array}{lll} \boldsymbol{p} > \boldsymbol{p}^+ & \boldsymbol{r} > \boldsymbol{r}^+ & \boldsymbol{f}_{l,r} > \boldsymbol{f}_{l,r}^+ \\ \boldsymbol{\tau} > \boldsymbol{\tau}^+ & \boldsymbol{w} > \boldsymbol{w}^+ & P_c > P_c^+ \end{array} \tag{5}
$$

P_r imposes penalty on difference between desired root link coordinates and simulated root link coordinates in the end of motion (goal condition). The difference considers hard maximum value $\boldsymbol{p}^+, \boldsymbol{r}^+$ and soft maximum value $\boldsymbol{p}^-, \boldsymbol{r}^-$ by using following barrier function:

$$
\begin{aligned} B(\boldsymbol{v}, \boldsymbol{v}^-, \boldsymbol{v}^+) &= \sum_{i \in [0,V)} B(v_i, v_i^-, v_i^+) \qquad (6) \\ & s.t. \quad \boldsymbol{v} = [v_0 \cdots v_{V-1}], \boldsymbol{v}^{\pm} = [v_0^{\pm} \cdots v_{V-1}^{\pm}] \\ B(v, v^-, v^+) &= \begin{cases} \|(v - v^-)/(v^+ - v^-)\|^2 & v > v^- \\ 0 & else \end{cases} \end{aligned}
$$

The barrier function B becomes 0 if first argument is smaller than soft max (second argument) and becomes 1 if first argument is equal to hard max (third argument). If first argument is larger than hard max, simulation is stopped and P_t penalty is considered. Similarly, P_f imposes penalty on foot contact force $\boldsymbol{f}$ and moment $\boldsymbol{m}$ and P_e imposes penalty on joint torque $\boldsymbol{\tau}$ and velocity $\boldsymbol{w}$ by using the same barrier function B. P_c imposes penalty on self collision and environment collision between environment and a set of links. In the following experiments, the set of links include head, torso and both arms. P_c is total number of collision while simulation. $W_{t,r,f,e,c}$ are constant values to weight each penalty. Concrete values used in the following experiments are listed in TABLE II, III.

C. Test of knee-contact motion generation

We solve following problem:

$$
minimize_{\boldsymbol{q}_1, t_m} \quad 1/S + P \tag{7}
$$

Joint angle trajectory is defined as follows (Eq (1)):

$$
\boldsymbol{q} = \boldsymbol{L}([\boldsymbol{q}_0, \boldsymbol{q}_1, \boldsymbol{q}_2]; [0, t_m/2, t_m], T(t)) \tag{8}
$$

1) Sitting down: We fix initial posture $\boldsymbol{q}_0$ and final posture $\boldsymbol{q}_2$ as shown in lower side of Fig. 11 (a). Search parameters are $\boldsymbol{q}_1$ and t_m. Initial guess of the parameters are uniform random in $\boldsymbol{q}_1 \in$ RoM (Range of Motion) of all joints and $t_m \in [1, 5]$ seconds. Additionally, we filter the joint angles to be symmetric posture.

2) Standing up: We use almost the same conditions as sitting down except for swapping initial postures for final postures as shown in lower side of Fig. 11 (c).

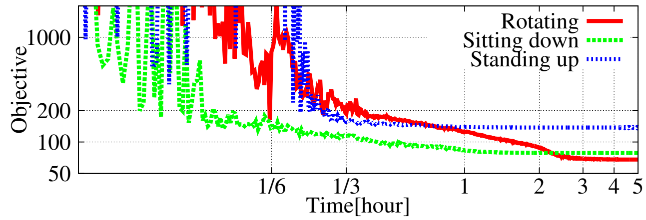


Fig. 3. Objective of evolutionary search (log scale)

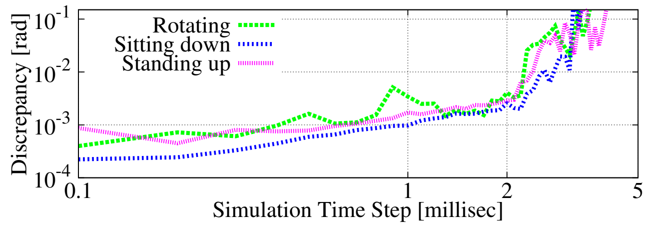


Fig. 4. Simulation accuracy and time step (log scale)

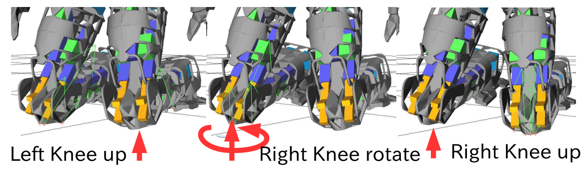


Fig. 5. Knee zoom snapshot of generated motion.

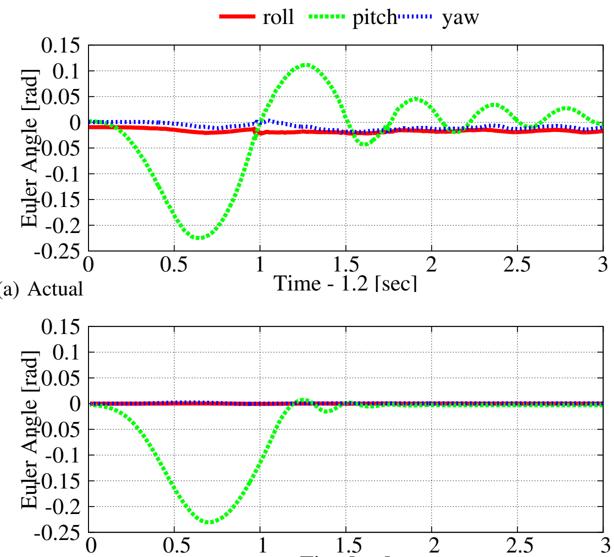
3) Rotating clockwise with knee contact: Initial posture and final posture are shown in lower side of Fig. 11 (b). The final posture is rotated 0.5 radian clockwise from the initial posture. The symmetric filter used for sitting down and standing up is not used for this experiment. Fig. 5 zooms knee links while rotating. We can confirm that complex contact states appear in the motion; First, left knee link moves upward. Second, right knee contact rotates keeping left knee link floating. Last, left knee link moves downward to ground and right knee link moves upward instead.

4) Computation time and Accuracy: We use 50 threads parallel computation in 5 hours. Total number of genes in each generation is 1024. Each gene is evaluated by 10 seconds simulation. The simulation speed is about 20 times faster than real world. Therefore, each generation evolves after (1024 genes $\times$ 10 seconds) / (20 kHz $\times$ 50 threads) $\approx$ 10 seconds. Fig. 3 shows transition of $Objective = 1/S + P$. We can confirm that all objectives become smaller than hard maximum penalty order (larger than 1000) and feasible motions are obtained. Although we search in 5 hours, objective functions of standing up and sitting down motions become almost constant after 1 hours. Fig. 4 shows simulation accuracy according to simulation time step. We expect a simulation result with small time step 10^{-5} [s] to be accurate, and discrepancy between the accurate result and a result with larger time step is used as accuracy evaluation of the larger time step. Similar evaluation can be found in [23]. The discrepancy we used is root mean square of time integration of difference of joint angles and root link attitude. Simulated results are similar if time step is smaller than 1 milliseconds. If time step is larger than 2 milliseconds, result becomes visibly different.

III. Experiments on Knee-contact motions

There exist large gaps between simulated motion and actual result due to error of model parameters. However, we confirmed that it was possible to achieve knee-contact motion due to large support region. Snapshots of all motions are shown in Fig. 11. Further, we compare actual results and simulated results to check the gaps in this section.

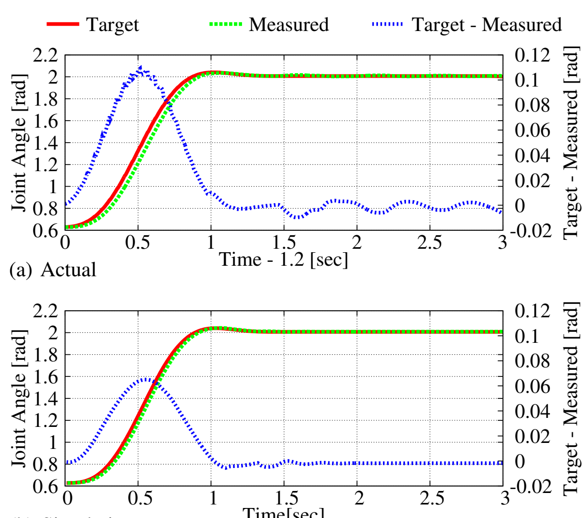
A. Sitting down



(a) Actual

(b) Simulation

Fig. 6. Root link attitude in world frame while sitting down motion



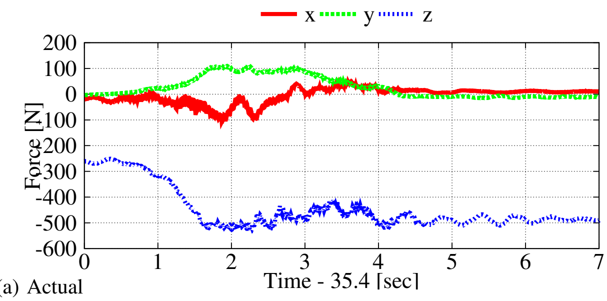
(a) Actual

(b) Simulation

Fig. 7. Left knee joint angles while sitting down motion

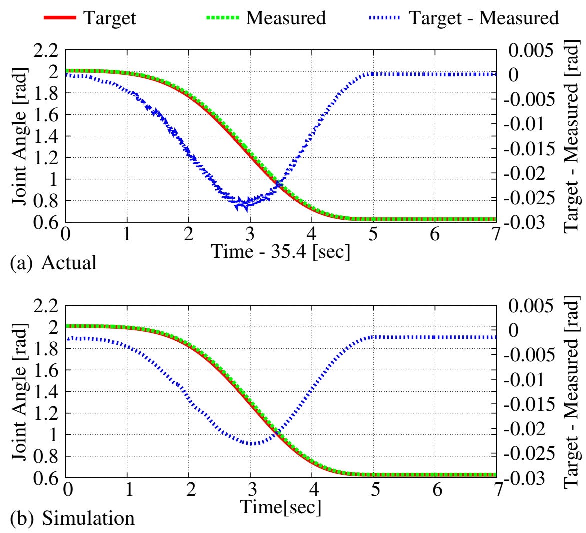
Upper side of Fig. 11 (a) shows actual results. Fig.6 compares actual root link attitude and simulated one. Actual results largely vibrate after sitting. By checking movie, timing of knee link contact seems to be fast and both feet temporary lift off. The feet lifting appears in simulated result although the lifting is smaller than real. We should check difference of mass parameters and contact points between actual and simulation world to reduce the gap. Fig. 7 shows target joint angle, measured one and difference between them (tracking error) of left knee joint. The tracking error reflects joint torque. We can confirm that actual and simulated results seem to be similar. However, the actual tracking error is over 30% larger than simulated result. We should check position controller to reduce the gap.

B. Standing up



(a) Actual

Fig. 8. 3axis force sensor of left foot while sitting down motion



(a) Actual

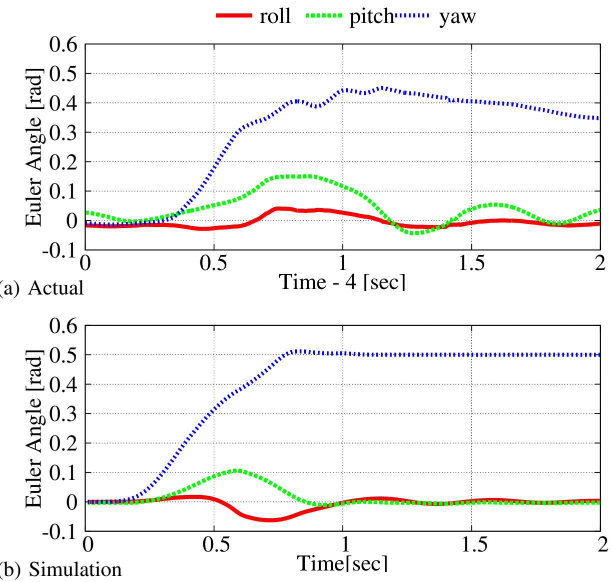
(b) Simulation

Fig. 9. Left knee joint angles while standing up motion

Upper side of Fig. 11 (c) shows actual results. Fig. 8 shows contact force of left foot. 'z' means vertical force. The force is 500 N after standing and 300 N while sitting. Therefore, about 200 N is applied to knee contact. Fig. 9 shows joint angles, measured ones and difference between them (tracking error) of left knee joint. In comparison with sitting down motion, the tracking error is 2 times larger because standing up motion needs more joint torque to sustain massive weight of humanoid robot. The reason why standing up motion is not reversed trajectory of sitting down motion is that our simulator considers dumping force for collision involving energy loss and the reversed trajectory is infeasible.

C. Rotating clockwise with knee contact

Upper side of Fig. 11 (b) shows actual results. Fig. 10 shows root link attitude. Actual attitude is estimated from acceleration sensor and gyro sensor by using Kalman filter, and simulated attitude is obtained by forward dynamics calculation. Because actual robot is not equipped with magnetic sensor, the estimated yaw rotation it not correct. However, we can confirm that actual robot largely rotates as simulated robot.



(a) Actual

(b) Simulation

Fig. 10. Root link attitude in world frame while rotating clockwise

IV. Summary and Conclusion

To investigate whole-body multi-contact behavior, we preliminarily research on knee-contact motion and show three experiments: Standing up from knee contact to foot contact, sitting down from foot contact to knee contact and rotating keeping knee link contact with ground. To generate knee-contact motion, we use goal-oriented simulation-based evolutionary search algorithm which does not specify detailed contact conditions to generate complex contact transition. Actually, generated knee-contact motions include complex contact states such as rotating around knee contact without given contact states. Further, we confirm that the generated motions can be achieved by actual life-size humanoid robot RHP4B. The contribution of this paper is to gain a foothold of whole-body multi-contact behavior by achieving knee-contact motions. We conclude that our approach will become increasingly important as robust robot systems enabling whole-body multi-contact behavior are developed more.

A. Future works

To improve performance of our approach, we have to overcome two issues. One is computational time. It took over 1 hour to generate knee-contact motions. However, our approach is beneficial to generate motion primitive as shown in this paper. Further, we try to generate not only whole-body trajectory but also controller with feedback to be used in wide range of applications. Actually, a paper about a walk controller will appear in IROS 2018 [19]. The other one is error of model parameters between actual and simulated world. We expand simulation-based model identification approach [24], and submit a paper as [25]. The expansion enables to achieve more dynamical and unstable motions.

References

[1] Y. Kakiuchi, M. Kamon, N. Shimomura, S. Yukizaki, N. Takasugi, S. Nozawa, K. Okada, and M. Inaba. Development of life-sized humanoid robot platform with robustness for falling down, long time working and error occurrence. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 689–696, 2017.

[2] R. Subburaman, J. Lee, D. G. Caldwell, and N. G. Tsagarakis. Online falling-over control of humanoids exploiting energy shaping and distribution methods. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 448–454, May 2018.

[3] S. Wang and K. Hauser. Realization of a real-time optimal control strategy to stabilize a falling humanoid robot with hand contact. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3092–3098, May 2018.

[4] S. Kajita, R. Cisneros, M. Benallegue, T. Sakaguchi, S. Nakaoka, M. Morisawa, K. Kaneko, and F. Kanehiro. Impact acceleration of falling humanoid robot with an airbag. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 637–643, Nov 2016.

[5] A. Hereid, E. A. Cousineau, C. M. Hubicki, and A. D. Ames. 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[6] K. Miura, F. Kanehiro, K. Kaneko, S. Kajita, and K. Yokoi. Slip-turn for biped robots. *IEEE Transactions on Robotics (TRO)*, Vol. 29, No. 4, pp. 875–887, 2013.

[7] K. Kojima, Y. Ishiguro, F. Sugai, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. Rotational sliding motion generation for humanoid robot by force distribution in each contact face. *IEEE Robotics and Automation Letters (RAL)*, Vol. 2, No. 4, pp. 2088–2095, Oct 2017.

[8] B. Henze, A. Dietrich, and C. Ott. An approach to combine balancing with hierarchical whole-body control for legged humanoid robots. *IEEE Robotics and Automation Letters (RAL)*, Vol. 1, No. 2, pp. 700–707, July 2016.

[9] S. Tonneau, A. Del Prete, J. Pettre, C. Park, D. Manocha, and N. Mansard. An efficient acyclic contact planner for multiped robots. *IEEE Transactions on Robotics (TRO)*, Vol. 34, No. 3, pp. 586–601, June 2018.

[10] S. Noda, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. Contact involving whole-body behavior generation based on contact transition strategies switching. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2787–2794, Oct 2015.

[11] H. Dai, A. Valenzuela, and R. Tedrake. Whole-body motion planning with centroidal dynamics and full kinematics. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 295–302, Nov 2014.

[12] M. Al Borno, M. de Lasa, and A. Hertzmann. Trajectory optimization for full-body movements with complex contacts. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 19, No. 8, pp. 1405–1414, Aug 2013.

[13] T. P. Runarsson and Xin Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, Vol. 4, No. 3, pp. 284–294, Sep 2000.

[14] S. G. Johnson. The nlopt nonlinear-optimization package. `http://ab-initio.mit.edu`.

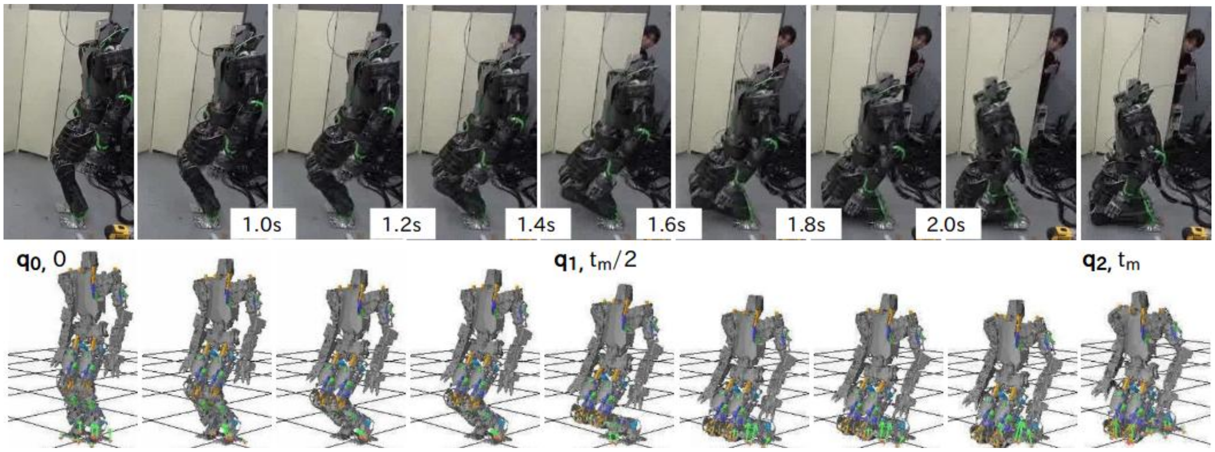
[15] R. Featherstone. *Rigid Body Dynamics Algorithms*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.

[16] M. Azad, V. Ortenzi, H. C. Lin, E. Rueckert, and M. Mistry. Model estimation and control of compliant contact normal force. In *IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 442–447, Nov 2016.

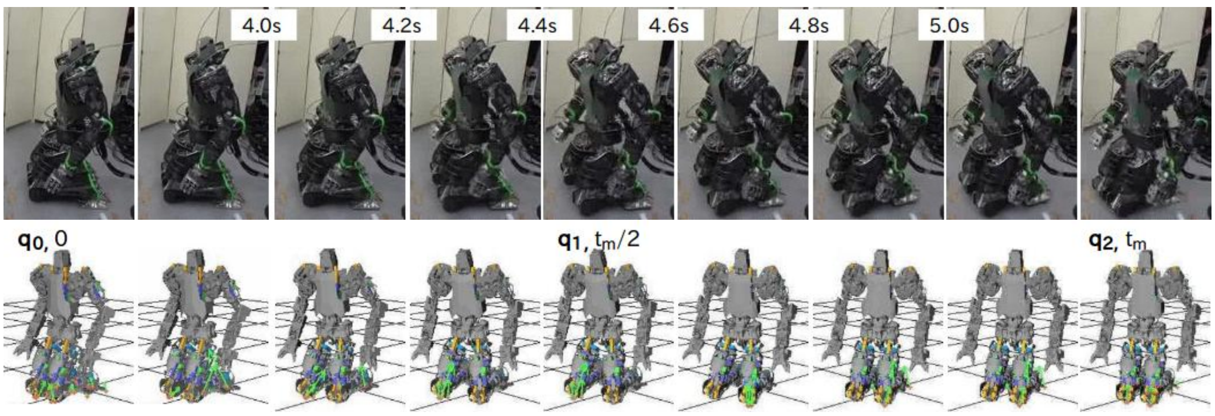
[17] M. A. Sherman, A. Seth, and S. L. Delp. Simbody: multibody dynamics for biomedical research. *Procedia IUTAM*, Vol. 2, pp. 241 – 261, 2011. IUTAM Symposium on Human Body Dynamics.

[18] K. Kojima, T. Karasawa, T. Kozuki, E. Kuroiwa, S. Yukizaki, S. Iwaishi, T. Ishikawa, R. Koyama, S. Noda, F. Sugai, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. Development of life-sized high-power humanoid robot jaxon for real-world use. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 838–843, Nov 2015.

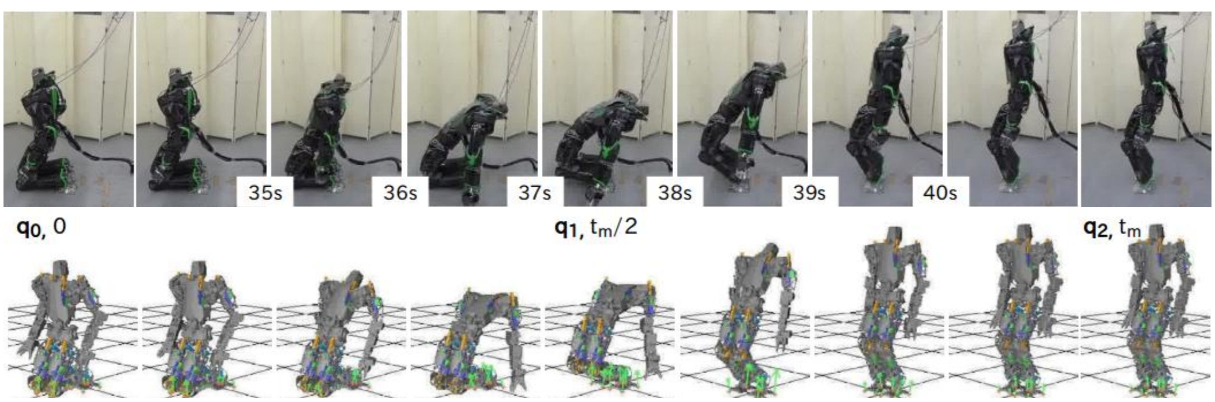
[19] K. N. K. Nguyen, S. Noda, Y. Kojio, F. Sugai, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. Robust and stretched-knee biped walking using direct joint angle control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1247–1254, Oct 2018.



(a) Sitting down in 1.45 [s]. Fast and dynamical collision between knee and grond appear.



(b) Rotating Clockwise in 1.06 [s]. Complex knee contact transition including sliding and rotating appear.



(c) Standing up in 4.99 [s]. Slower than sitting down motion due to high load on knee joints.

Fig. 11. Snapshots of knee contact motions. Upper side shows actual experiment and lower side shows simulation result.

[20] P. Fernbach, S. Tonneau, and M. Taïx. CROC: Convex Resolution Of Centroidal dynamics trajectories to provide a feasibility criterion for the multi contact planning problem. preprint in https://hal.archives-ouvertes.fr/hal-01726155, May 2018.

[21] S. Lengagne, J. Vaillant, E. Yoshida, and A. Kheddar. Generation of Whole-body Optimal Dynamic Multi-Contact Motions. *The International Journal of Robotics Research (IJRR)*, Vol. 32, No. 9-10, pp. 1104–1119, 2013.

[22] R. Terasawa, S. Noda, K. Kojima, R. Koyama, F. Sugai, S. Nozawa, Y. Kakiuchi, K. Okada, and M. Inaba. Achievement of dynamic tennis swing motion by offline motion planning and online trajectory modification based on optimization with a humanoid robot. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pp. 1094–1100, Nov 2016.

[23] T. Erez, Y. Tassa, and E. Todorov. Simulation tools for model-based robotics: Comparison of bullet, havok, mujoco, ode and physx. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4397–4404, May 2015.

[24] J. Tan, Z. Xie, B. Boots, and C. K. Liu. Simulation-based design of dynamic controllers for humanoid balancing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2729–2736, Oct 2016.

[25] S. Noda, F. Sugai, K. Kojima, K. N. K. Nguyen, Y. Kakiuchi, K. Okada, and M. Inaba. Semi-passive walk and active walk by one bipedal robot. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, Nov 2018.