Development of a Bipedal Robot that Walks Like an Animation Character

Seungmoon Song[1], Joohyung Kim[2], and Katsu Yamane[2]

***Abstract*— Our goal is to bring animation characters to life in the real world. We present a bipedal robot that looks like and walks like an animation character. We start from animation data of a character walking. We develop a bipedal robot which corresponds to lower part of the character following its kinematic structure. The links are 3D printed and the joints are actuated by servo motors. Using trajectory optimization, we generate an open-loop walking trajectory that mimics the character's walking motion by modifying the motion such that the Zero Moment Point stays in the contact convex hull. The walking is tested on the developed hardware system.**

I. INTRODUCTION

Creating robots that embody animation characters in the real world is highly demanded in the entertainment industry because such robots would allow people to physically interact with characters that they have only seen in films or TV. To give a feeling of life to those robots, it is important to mimic not only the appearance but also the motion styles of the characters. The process is not straightforward, since most animation characters are not designed to be designed and animated considering physical feasibility and their motions are seldom physically correct.

Animation characters have evolved to be more realistic. Using computer graphic techniques, we can design 3D characters, and generate more natural and physically plausible motions with them. Among many other motions, due to the interest in bipedal characters, generating realistic and natural bipedal walking has been extensively studied by many researchers [1]–[5]. One approach is to directly explore the walking motions with trajectory optimization [6] to find desired motions which obey the physics law [1], [2]. Another approach is to develop walking controllers that allow characters to walk in physics-based simulation [3]–[5]. Nowadays, the computer graphics community is establishing techniques to animate mechanical characters [7]–[10]. Recently, a computational framework of designing mechanical characters with desired motions has been proposed [8], [9].

In the meantime, the desire of having lifelike bipedal walking in real world has persisted in the field of robotics over decades [11]–[14]. For the goal of solving real world problems, such as helping elders' daily life or resolving natural and man-made disasters, humanoids with high-fidelity control of joint positions and torques have been developed [12], [15], [16]. On the other hand, more compact bipedal robots for entertainment and hobby have been developed using servo motors [17], [18]. More recently, miniature bipedal robots with servo motors and 3D printed links are gaining attention [19], [20].

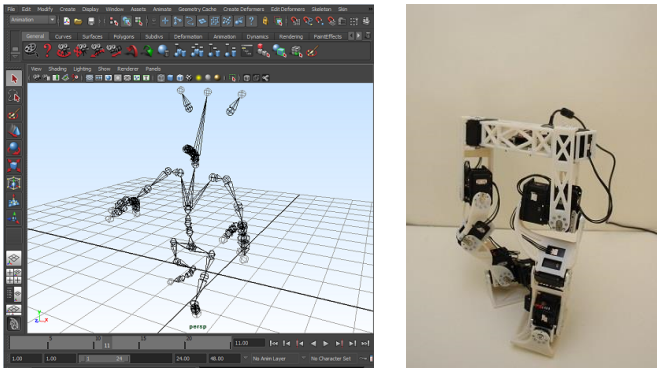

Fig. 1. Animation character and robot. Our goal is to develop a bipedal robot that looks like and walks like an animation character.

Here, we develop a bipedal robot that is designed after an animation character, and generate a character-like walking motion for the robot (Fig. 1). The kinematic structure of the robot is designed based on the animation character, and is compact enough to be covered by the character suit (Sec. III). The links are 3D printed and the joints are actuated by servo motors. For character-like walking motion, we generate an open-loop walking trajectory resorting to trajectory optimization (Sec. IV). The objective of the walking trajectory is to be similar to the character's walking, while preserving stability by insuring the Zero Moment Point (ZMP) criterion. The optimized walking is tested on the hardware and generates stable walking. Through the hardware experiments we observe discrepancy between the simulation and hardware walking which limits the hardware from walking at optimized speed, and we discuss how we plan to resolve this problem (Sec. V).

II. FRAMEWORK FOR DEVELOPING HARDWARE AND WALKING TRAJECTORY

The main challenge of this project comes from the fact that the original animation character and its motions are not designed considering physical constraints. For example, in our target character the ankle-foot section consists of three joints, where each has 3 degrees of freedom (DOF), and integrating nine actuators in a small foot is not practical. Moreover, the walking motions in animation are generated with key frames that artists crafted, which is not physically

[1]S. Song is with the Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA smsong@cs.cmu.edu

[2]J. Kim and K. Yamane are with Disney Research, Pittsburgh, 4720 Forbes Ave. Suite 110, Pittsburgh, PA 15213, USA {joohyung.kim, kyamane}@disneyresearch.com

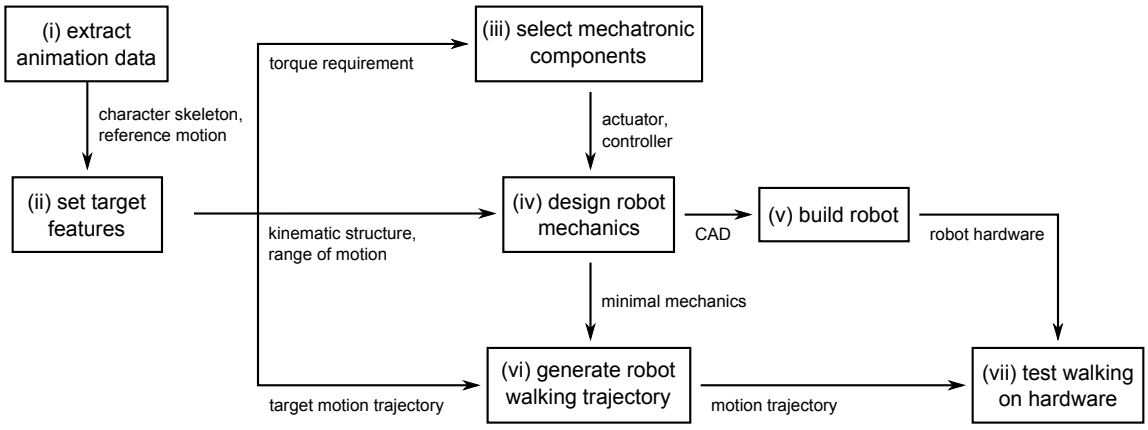

Fig. 2. Framework for designing mechanics and walking trajectory.

realizable. In other words, if we just playback the walking motion in the animation on a real robot, it will fall down.

Figure 2 gives the overview of how we solve this problem. To develop the robot, we first extract and analyze animation data (i). Based on the analysis, we set target features such as the kinematic structure of the robot, range of motion, and torque requirements of each joint (ii). Then, we select mechatronic components such as actuators that can realize the target features (iii), design the segments (iv), and build the robot (v).

To generate an open-loop walking trajectory for the robot, we first modify the animation walking motion to be suitable for the robot (i-ii). Specifically, we map the motion to the robot's configuration space, and modify the motion to keep the stance foot flat on the ground so that stability is guaranteed by the traditional ZMP criterion [21], which is widely used for bipedal walking [12], [22], [23]. To generate a walking motion that is stable while preserving the original walking style, we resort to trajectory optimization [6], [24] with the objective function of minimizing the deviation from the target walking motions and keeping the center of pressure (COP) or the ZMP in the support polygon (vi). Finally, we test the optimized walking motion on real hardware by tracking the joint trajectories (vii).

We extract the original animation data from an animation software MAYA, then process and optimize it using MATLAB Simulink/SimMechanics (R2013a).

III. ROBOT HARDWARE OF THE CHARACTER LOWER BODY

A. Extract Data from Animation

We extract the skeleton structure of the animation character from a given Maya. Each leg has 15 DOF; 3 DOF at the hip, 3DOF at the knee, 3DOF at the ankle, and 6DOF in the foot. Foot, shank and thigh heights and pelvis width are reported in Table I. Comparing to typical miniature bipedal robots, the pelvis is wide (18.8 cm) relative to the leg length (20 cm = 8.1+8.9+3.0 cm), which makes the lateral balance during walking more challenging.

B. Set Target Features

The skeleton data of the animation character does not include the inertia properties. Moreover, it is impractical to build a hardware exactly as the character. For example, the ankle-foot part consists of 9DOF, which is hard to implement in the volume of the character's foot. Considering these factors, we have had four criteria for designing the robot hardware:

- set the number of DOF realizable,
- keep dimensions close to the animation character,
- insure the range of motion (ROM) for walking,
- and insure the torque capability for walking.

Since our goal is to mimic the walking motion of the original animation, we investigated the dynamics of the target walking motion in simulation, where the detail is later explained in Sec. IV-C. (A model of a robot including the inertia properties is required to run the simulation. Since the animation data does not include the properties, they are determined through an iterative process that involves running the simulation and designing the robot.) Through the simulation study, we have verified that a leg configuration of typical miniature humanoids, which consists a thigh, a shank, and a foot connected by a 3DOF hip joint, a 1DOF knee joint, and a 2DOF angle joint, can mimic the walking motion. Furthermore, the simulation result provided an estimate of the range of motion (Table I) and torque requirements of each joint for the walking motion.

C. Select Mechatronic Components

We have searched for joint actuators based on the estimated torque requirements. We limited the search on servo motors, since we devise to track joint trajectories. We have selected Dynamixel servos which are widely used for humanoids with joint position controllers [17], [18], [25]. More specifically, we use MX-106T for the hip and knee joints and MX-64T for the ankle joints [26]. The maximum torques MX-106T and MX-64T can exert ($\tau_{106,max}$ Nm and $\tau_{64,max}$ Nm) given the angular velocities (ω rad s^{-1}) are

$$\begin{aligned} \tau_{106,max}(\omega) &\approx 8.81 - 1.80 \times |\omega| \\ \tau_{64,max}(\omega) &\approx 4.43 - 0.66 \times |\omega| \end{aligned} \quad (1)$$

TABLE I

TARGET AND FINAL KINEMATIC CONFIGURATION. THE TARGET SEGMENT DIMENSIONS ARE FROM THE ANIMATION CHARACTER, AND THE ROM REQUIREMENTS ARE FROM THE SIMULATION STUDY.

dim.	full body (height)	pelvis (width)	thigh (height)	shank (height)	foot (height)	foot (length)
anim.	73.0	18.8	8.1	8.9	3.0	10.0
robot	-	20.0	8.9	9.8	3.7	10.6
						(unit: cm)

ROM	hip_{yaw} (intern ∼extern)	hip_{roll} (adduct ∼abduct)	hip_{pitch} (extend ∼flex)	knee_{pitch} (extend ∼flex)	ankle_{pitch} (extend ∼flex)	ankle_{roll} (invert ∼evert)
req.	-15∼45	-35∼15	-15∼60	20∼115	0∼70	0∼20
robot	-40∼90	-55∼90	-105∼135	0∼115	-30∼120	-20∼90
						(unit: degree)

which are set as constraints in the walking motion optimization (Sec. IV-C and Sec. IV-D).

We use OpenCM9.04 microcontroller board with OpenCM458EXP expansion board [26], and send joint position commands to the servos every 10 ms in TTL. The power is off-board.

D. Design Robot Mechanics

The kinematic structure were designed to keep the dimensions close to the animation character, and to insure the ROM for walking (Fig. 3-(a,b)). The order of the actuators from medial to distal is hip yaw, hip roll, hip pitch, knee pitch, ankle pitch, and knee roll. The thigh, shank and foot segments of the robot are slightly longer than the animation character to secure the ROM, and the pelvis width is wider to keep the proportion similar (Table I). As in the animation character, the three rotational axes of the hip joints and the two rotational axes of the ankles are co-aligned at each joint. In addition, we have intended to align the hip, knee and ankle pitch joint in the sagittal plane at the default configuration (stand straight configuration). However, due to the size of the ankle servos, the ankle pitch joint is placed forward (Fig. 3-(b)). The knee servos are tilted at default configuration to realize the target ROM.

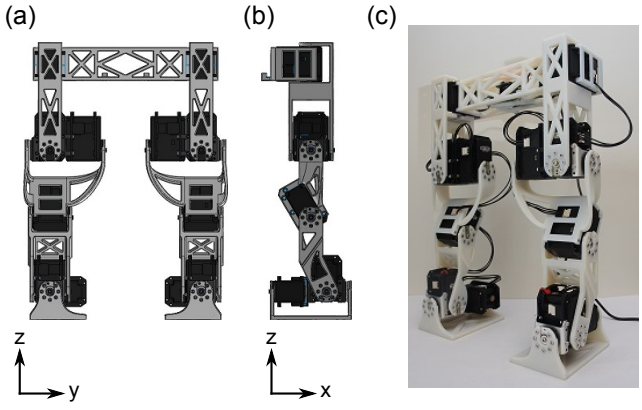

Fig. 3. Hardware of the robot. (a) and (b) show the front and side view of the CAD of the robot, and (c) shows the actual hardware.

TABLE II

HARDWARE SPECIFICATION.

height	35 cm	DOF	12	power	off-board
weight	2.55 kg	actuator	MX-64T (×8), MX-106T (×4)	controller	OpenCM9.04, OpenCM458EXP
leg len	22 cm				
pel wid	20 cm	segment	RGD525	comm prot	TTL

(leg len: leg length, pel wid: pelvis width, ctrl prot: control protocol)

E. Build Robot

We 3D print all the segments, and assemble the robot with the segments, aluminum frames, and the actuators. Specifically, we use Stratasys' Objet260 Connex [27] to 3D print the segments with RGD525, and use Robotis' aluminum frames. The mechanics and the electronics are summarized in Table II.

IV. STABLE WALKING MOTION THAT MIMICS THE CHARACTER'S WALKING

A. Extract Data from Animation

As most animations, the original walking motions given to us are handcrafted by artists with an animation software (Maya in our case). Specifically, we have two similar, but not identical, asymmetric gaits, from which we can design four different single stance motions. For the animation walking motions, one stride takes 1.1 to 1.2 seconds; a double support phase compose about 15% of a full step; a stance phase starts with a heel-strike pose and ends with a toe-off pose; and the foot is rotated externally about 30 degrees during stance (Fig. 5-(a)).

We aim to generate one walking motion for the robot, which *looks* similar to the animation. Since the robot has different kinematic configuration with the animation character, rather than the joint trajectories, we extract the position trajectories of the segments in the Euclidean space $\bar{\boldsymbol{p}}_{anim}$. In detail, we extract the positions of the center of the pelvis, hips, knees, ankles, and tip of toes, and the motions are provided with a time frequency of 24 frames per second.

B. Set Target Features

At this step, we generate the target joint trajectories of the single stance motion for the robot, $\bar{\boldsymbol{\varphi}}_{tgtSS}$, based on the original motion in the Euclidean space, $\bar{\boldsymbol{p}}_{anim}$. We do not generate target motions for the double support phase, since it is relatively short and there is not much freedom in the motion since both feet remains at their positions. For the single stance motion, we modify the animation motion to be suitable for the robot by 1) keeping the stance foot flat on the ground, and 2) mapping the motion in the Euclidean space into the robot's configuration space. In addition, based on the original motions and the range of motion of the robot, we set the stance foot to point outward by 30 degrees, and set the step length and step width to be 15 cm and 13 cm, respectively.

The original single stance motion, $\bar{\boldsymbol{p}}_{anim,SS}$, is first modified in the Euclidean space to keep the stance foot flat on its position ($\bar{\boldsymbol{p}}_{tgt,SS}$). The stance leg in all time frames is linearly

translated to locate the stance foot at its position. The swing leg is translated only in the horizontal plane, and its joint positions are scaled along the vertical (or gravitational) z-axis to preserve the original vertical motion of the swing foot while maintaining the relative positions between the stance hip and the swing hip. The swing leg trajectory is further modified so that the swing foot starts and ends at positions that connects with the previous and next stance foot configurations, respectively.

To generate target motion trajectories for the robot, we convert $\bar{\boldsymbol{p}}_{tgtSS}$, which are target trajectories of the segments in the Euclidean space, into the robot's configuration space. This process differ from typical inverse kinematics (IK), in that there are target positions for each segments (opposed to having a target position only for one end effector) and these targets are not always reachable by the robot [28]. We solve this by running physics simulation for each pose with virtual spring dampers connecting the segments to their target positions. By reading joint angles from the simulation results of all poses, we obtain the target joint trajectories $\bar{\boldsymbol{\varphi}}_{tgtSS}$ (Fig. 5-(b)).

By modifying the four different animation walking motions with the same process, we acquire four target joint trajectories for the single stance phase. All four are shown in Figure 4 in gray dotted lines.

C. Generate robot walking trajectory (single stance phase)

Figure 5-(b) shows one of the target motions $\bar{\boldsymbol{\varphi}}_{tgtSS}$ tracked in physics simulation by assuming the stance foot is welded to the ground. The simulation shows that the COP (black line) gets out of the support polygon (gray area), which means that the robot will fall down if we track this motion with the robot. The remaining work is to generate a walking motion which are similar to the four sets of $\bar{\boldsymbol{\varphi}}_{tgtSS}$'s and is physically reliable without falling down, i.e. COP remains in the foot as known as the ZMP criterion [21]. We solve this problem with trajectory optimization.

Trajectory optimization is a well established method for designing optimal trajectories for given criteria and constraints [6], [24]. We parametrize the single stance motion as $\hat{\boldsymbol{\varphi}}_{SS}$ with 145 ($=12\times12+1$) parameters, where 12 parameters represents equally time-spaced nodes of each joint trajectories ($\times12$), and an additional parameter defines the time duration of the single stance motion ($+1$). To evaluate a set of parameters $\hat{\boldsymbol{\varphi}}_{SS}$, we reconstruct joint trajectories from the parameters, solve the forward kinematics (FK) to investigate the motion, and solve the inverse dynamics (ID) to examine the dynamics. In more detail, we reconstruct joint trajectories with spline interpolation, then run a simulation in joint angle tracking mode with the stance foot welded on the ground to solve FK and ID. From the results of FK and ID, we evaluate the single stance motion, based on how similar the motion is to the target motion ($\left\|\hat{\boldsymbol{\varphi}}_{SS},\bar{\boldsymbol{\varphi}}_{tgtSS}\right\|$) and how close the COP remains at the center of the stance foot ($\|\boldsymbol{COP}_{SS}\|$). The difference of the motions and the COP deviation are defined as the root mean squares of the differences of the joint angles as $\left\|\hat{\boldsymbol{\varphi}}_{SS},\bar{\boldsymbol{\varphi}}_{tgtSS}\right\| = \sqrt{\hat{\varphi}^2_{SS,traj}-\bar{\varphi}^2_{tgtSS,traj}} + \sum_{joint}\sum_k \sqrt{\hat{\varphi}^2_{SS,t}-\bar{\varphi}^2_{tgtSS,t}}$, and $\|\boldsymbol{COP}_{SS}\| = \sum_t \sqrt{x^2_{COP,SS}+y^2_{COP,SS}}$, respectively. Note that the former one is calculated at every target frames, k, and the latter one is calculated at every simulation time step, t.

Since we can evaluate a parametrized single stance motion, $\hat{\boldsymbol{\varphi}}_{SS}$, we can formulate the problem of generating the optimal motion as a nonlinear constrained optimization problem:

$$
\begin{aligned}
\min_{\hat{\boldsymbol{\varphi}}_{SS}} \quad & c_{SS}\left\|\hat{\boldsymbol{\varphi}}_{SS},\bar{\boldsymbol{\varphi}}_{tgtSS}\right\| \\
& +c_{COP}\left\|\boldsymbol{COP}_{SS}\right\| \qquad (2)\\
\text{subject to} \quad & \boldsymbol{p}^{foot}_{SS,k=1,K} = \boldsymbol{p}^{foot}_{tgtSS,k=1,K} && -C1\\
& \|\boldsymbol{COP}_{SS},\boldsymbol{COP}_{limit}\| < 0 && -C2\\
& z^{Sw-foot}_{SS,k=2,\cdots K-1} > 0 && -C3\\
& \boldsymbol{\tau} < \boldsymbol{\tau}_{max}(\boldsymbol{\omega}) && -C4\\
& \left|\boldsymbol{GRF}_z\right| > 0 && -C5\\
& \left|\boldsymbol{GRF}_{x,y}/\boldsymbol{GRF}_z\right| < \mu && -C6.
\end{aligned}
$$

The goal is to minimize the deviation from the target motion $\hat{\boldsymbol{\varphi}}_{SS}$, while increasing the stability of the motion, i.e. keeping the COP near the center of the stance foot. The weighting coefficients c_{SS} and c_{COP} are heuristically found to balance these goals. $C1$ guarantees that the swing foot starts and ends at target positions, which is necessary for symmetric walking, and all other constraints ($C2\sim C6$) insures physically reliable walking motion: $C2$ insures that the model does not fall down, $C3$ keeps the swing foot from scuffing by constraining the height of the swing foot ($z^{Sw-foot}_{SS,k=2,\cdots K-1}$) to be higher than the ground, $C4$ checks if the servos are capable of generating such torques (Eq. (1)), $C5$ assures the stance foot does not take off the ground, and $C6$ is intended to avoid slipping at the stance foot ($\boldsymbol{GRF}_{x,y,z}$ are the ground reaction forces along each axis, we use a friction coefficient $\mu = 0.6$). In theory, $C2$ is enough to prevent the robot from falling down without the cost term of COP_{SS}. However, by keeping the COP near the center of the foot, we attempt to compensate the modeling errors of the simulation and small disturbances which always exist in real environment.

To optimize the trajectory parameters, $\hat{\boldsymbol{\varphi}}_{SS}$, we use covariance matrix adaptation evolution strategy (CMA-ES, [29]), which is widely used for optimizing policy based walking controllers [30], [31]. To force the constraints, we add a large constant $c_L = 10^6$ to the cost when the constraints are violated. We use one of the target trajectories as the initial parameters. Our CMA-ES runs with a population size of 64 for 6000 generations. Without optimizing our implementation for running speed, it takes about 2 days on a 3.4G Hz CPU desktop.

D. Generate robot walking trajectory (double stance phase)

Once we optimized the single stance motion, $\hat{\boldsymbol{\varphi}}_{optSS}$, to generate the full walking motion, we generate the double stance phase motion, $\hat{\boldsymbol{\varphi}}_{DS}$, which connects the last pose of single stance phase and the first pose of the next single stance phase. We interpolated the segment positions in the Euclidean space between the target start and end poses,

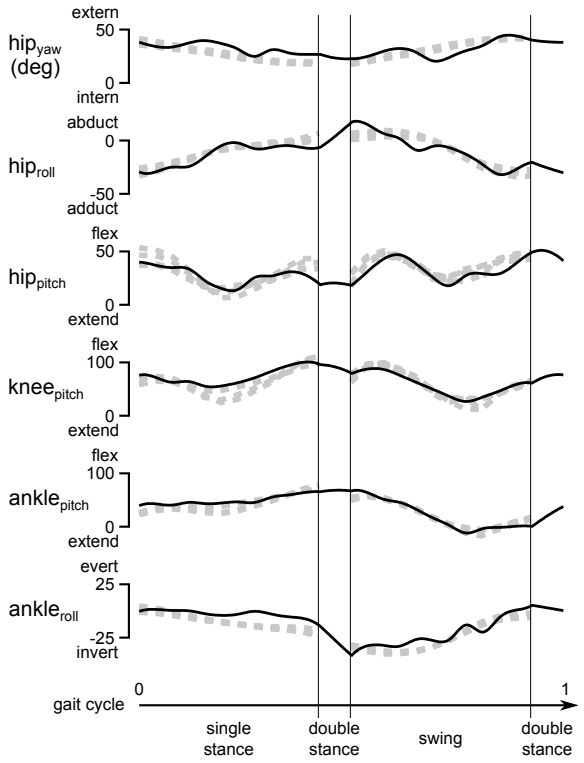

Fig. 4. Optimized walking trajectory. The optimize trajectory for a full stride (black solid line) are compared to the target joint trajectories modified from the animation data (gray dotted lines).

then solve the IK to get the joint trajectories. In our case, the interpolated double stance motion is stable, where the COP remains in the support polygon progressing from the previous stance foot toward the next stance foot. If the motion is not stable, we can generate a stable one by optimizing the trajectory similar as for the single stance phase. For optimizing the double support motion, the cost will be $\|\boldsymbol{COP}_{SS}\|$, since we do not have a target double stance motion, $C1$ will be applied for all time steps, and COP_{limit} will be the support polygon which covers both feet.

We obtain the full walking motion by connecting the single stance motion and the double stance motion. The joint trajectories of a full stride of one leg, from single stance, double stance, swing, and double stance are shown in Figure 4, along with the target single stance and swing motion. The ankle roll trajectory deviates most from the target trajectories. It is because the ankle can modulate the COP most effectively with small movements. The resulting walking motion and its COP and COM trajectories are shown in Figure 5-(c).

E. Test walking on hardware

We test the optimized walking trajectory on hardware by tracking the open-loop joint trajectories with the servo motors. When we play back the optimized trajectory, the robot wobbles forward. It is because the robot does not produce the motion perfectly. For example, the stance leg flexes more and scuffs the swing foot at the beginning and end of the swing phase. This causes the swing foot to push against the ground and the stance foot to slip, which results in unstable walking. We verified that the robot produces different motions under load, by comparing the walking to sky-walking (playing the walking motion in the air), and speculate two sources of motion error: the deformations of the links and the error of the servo motors.

We observed that the robot slips less as we play back the optimized motion slower, and the resulting walking looks closer to the optimized walking. In theory, the COP trajectory will be closer to the COM trajectory for slower walking. The optimized COM trajectory (red line in Fig. 5-(c)) mostly remains above the stance foot and the unstable moment is short. Furthermore, when the COM is out of the stance foot it progresses toward either the current or the next stance foot; therefore, the instability may be compensated by the next stable phase. Figure 5-(d) shows the robot walking two times slower than the optimized one.

V. FUTURE DIRECTION

We plan to focus on improving the system so that it better tracks optimal trajectories. First, we can generate optimal trajectories that are easier to track. For example, when optimizing a trajectory we can include a cost term regarding the deformation of the linkages. In addition, as we optimize for the robustness against COP deviation, we can consider the robustness against swing foot height. Second, better segments can reduce the deformations. We can investigate materials and structural designs for stiffer segments. Also, improving the strength of the segments is essential since the segments occasionally broke when the robot fell down. Third, better tracking may be achieved by improving the servo control of the motors. Currently, we use the default PID gains of the Dynamixels, which we can adjust through experiments. Furthermore, we can add a feedforward controller which adds offsets to the angular trajectories, considering the load the joints and the segments will bear.

REFERENCES

[1] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 43, 2012.

[2] M. Al Borno, M. De Lasa, and A. Hertzmann, "Trajectory optimization for full-body movements with complex contacts," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 19, no. 8, pp. 1405–1414, 2013.

[3] K. Yin, K. Loken, and M. van de Panne, "Simbicon: Simple biped locomotion control," in *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3. ACM, 2007, p. 105.

[4] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Optimizing walking controllers for uncertain inputs and environments," *ACM Transactions on Graphics (TOG)*, vol. 29, no. 4, p. 73, 2010.

[5] T. Geijtenbeek, M. van de Panne, and A. F. van der Stappen, "Flexible muscle-based locomotion for bipedal creatures," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 6, p. 206, 2013.

[6] A. Witkin and M. Kass, "Spacetime constraints," in *ACM Siggraph Computer Graphics*, vol. 22, no. 4. ACM, 1988, pp. 159–168.

[7] L. Zhu, W. Xu, J. Snyder, Y. Liu, G. Wang, and B. Guo, "Motion-guided mechanical toy modeling." *ACM Trans. Graph.*, vol. 31, no. 6, p. 127, 2012.

[8] S. Coros, B. Thomaszewski, G. Noris, S. Sueda, M. Forberg, R. W. Sumner, W. Matusik, and B. Bickel, "Computational design of mechanical characters," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, p. 83, 2013.

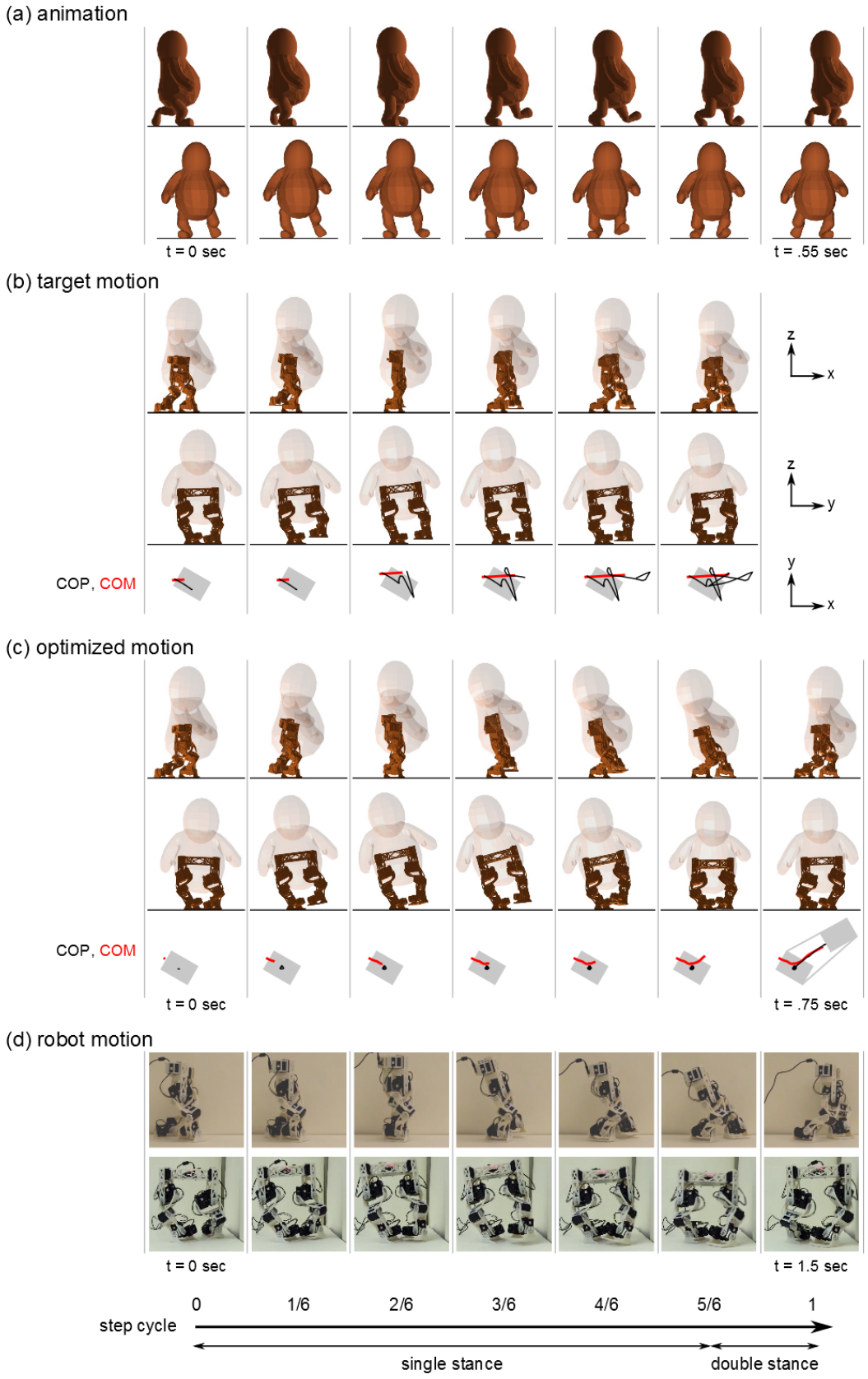

Fig. 5. Walking motions. The goal is to make the robot walk similar to the walking motion in animation which is shown in (a). We first modify the animation walking as (b) by mapping the motion to the robot configuration space and by keeping the stance foot flat on the ground. Since the modified motion is physically incorrect (the COP gets out of the foot in (b)), we optimize the walking to be physically realizable by keeping the COP inside the support polygon as (c). Finally, as shown in (d) we play back the optimized walking motion on the real hardware. Due to the deformation of the hardware, which results in excessive slipping at the optimized motion speed, we play back the motion two times slower than the optimized speed.

[9] B. Thomaszewski, S. Coros, D. Gauge, V. Megaro, E. Grinspun, and M. Gross, "Computational design of linkage-based characters," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 64, 2014.

[10] M. Bächer, E. Whiting, B. Bickel, and O. Sorkine-Hornung, "Spin-it: optimizing moment of inertia for spinnable objects," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 96, 2014.

[11] J.-I. Yamaguchi, A. Takanishi, and I. Kato, "Development of a biped walking robot compensating for three-axis moment by trunk motion," in *Intelligent Robots and Systems' 93, IROS'93. Proceedings of the 1993 IEEE/RSJ International Conference on*, vol. 1. IEEE, 1993, pp. 561–566.

[12] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent asimo: System overview and integration," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3. IEEE, 2002, pp. 2478–2483.

[13] M. H. Raibert *et al.*, *Legged robots that balance*. MIT press Cambridge, MA, 1986, vol. 3.

[14] K. Kaneko, F. Kanehiro, M. Morisawa, K. Miura, S. Nakaoka, and S. Kajita, "Cybernetic human hrp-4c," in *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*. IEEE, 2009, pp. 7–14.

[15] J. Kim, Y. Lee, S. Kwon, K. Seo, H. Kwak, H. Lee, and K. Roh, "Development of the lower limbs for a humanoid robot," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4000–4005.

[16] X. Xinjilefu, S. Feng, and C. G. Atkeson, "Dynamic state estimation using quadratic programming," in *Intelligent Robots and Systems (IROS), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 989–994.

[17] J. Han and D. Hong, "Development of a full-sized bipedal humanoid robot utilizing spring assisted parallel four-bar linkages with synchronized actuation," in *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2011, pp. 799–806.

[18] I. Ha, Y. Tamura, H. Asama, J. Han, and D. W. Hong, "Development of open humanoid platform darwin-op," in *SICE Annual Conference (SICE), 2011 Proceedings of*. IEEE, 2011, pp. 2178–2181.

[19] M. Lapeyre, P. Rouanet, and P.-Y. Oudeyer, "The poppy humanoid robot: Leg design for biped locomotion," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 349–356.

[20] B. D. Johnson, "Jimmy: The robot that changed it all," *Computer*, vol. 46, no. 3, pp. 0104–107, 2013.

[21] M. Vukobratović and B. Borovac, "Zero-moment pointthirty five years of its life," *International Journal of Humanoid Robotics*, vol. 1, no. 01, pp. 157–173, 2004.

[22] J.-Y. Kim, I.-W. Park, and J.-H. Oh, "Walking control algorithm of biped humanoid robot on uneven and inclined floor," *Journal of Intelligent and Robotic Systems*, vol. 48, no. 4, pp. 457–484, 2007.

[23] S. Song, Y.-J. Ryoo, and D. W. Hong, "Development of an omnidirectional walking engine for full-sized lightweight humanoid robots," in *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, 2011, pp. 847–854.

[24] C. R. Hargraves and S. W. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 4, pp. 338–342, 1987.

[25] O. Ly, M. Lapeyre, and P.-Y. Oudeyer, "Bio-inspired vertebral column, compliance and semi-passive dynamics in a lightweight humanoid robot," in *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. IEEE, 2011, pp. 1465–1472.

[26] Robotis. (2014) Mx-106 e-manual. [Online]. Available: http://http://support.robotis.com/en/home.htm

[27] Stratasys. (2014) Object260 connex. [Online]. Available: http://www.stratasys.com/3d-printers/design-series/objet260-connex

[28] S. R. Buss, "Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods," *IEEE Journal of Robotics and Automation*, vol. 17, pp. 1–19, 2004.

[29] N. Hansen, "The cma evolution strategy: A comparing review," *Towards a New Evolutionary Computation. Advances on Estimation of Distribution Algorithms. Springer*, p. 75.102, 2006.

[30] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Optimizing walking controllers," in *ACM Transactions on Graphics (TOG)*, vol. 28, no. 5. ACM, 2009, p. 168.

[31] S. Song and H. Geyer, "A neural circuitry that emphasizes spinal feedback generates diverse behaviors of human locomotion," *The Journal of physiology*, in revision.