Generation of Walking Motions Based on Whole-Body Poses and QP Control

Raphael Grimm, Abderrahmane Kheddar, Tamim Asfour

▶ **To cite this version:**

Raphael Grimm, Abderrahmane Kheddar, Tamim Asfour. Generation of Walking Motions Based on Whole-Body Poses and QP Control. Humanoids 2018 - 18th IEEE-RAS International Conference on Humanoid Robots, Nov 2018, Beijing, China. pp.510-515, 10.1109/HUMANOIDS.2018.8624913 . lirmm-03131958

HAL Id: lirmm-03131958

https://hal-lirmm.ccsd.cnrs.fr/lirmm-03131958

Submitted on 4 Feb 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

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

Generation of walking Motions based on Whole-Body Poses and QP Control

Raphael Grimm[1], Abderrahmane Kheddar[2] and Tamim Asfour[1]

***Abstract*— Generating and executing whole-body motions for humanoid robots remains a challenging research question. In this paper, we present an approach that combines human motion data and QP-based control to generate humanoid motion. Following the contacts-before-motion paradigm, we first generate a sequence of stances based on our previous work on data-driven generation of whole-body multi-contact pose sequences from human motion data and their mapping to the target robot kinematics. In this paper, we address the next step of closed-loop execution of stance sequences based on QP controllers. We evaluated the approach in simulation on the humanoid robot ARMAR-4 and HRP4. The results show that our approach can successfully execute stance sequences generated by our previous work and thus the viability of learning locomotion patterns from human demonstrations.**

I. Introduction

Locomotion in an unstructured environment is an ability necessary for bipedal humanoid robots to interact with the environment in a real-world scenario, i.e. in a household or outdoors. In our previous work [1] we concentrated on the autonomous detection of end-effector contact opportunities in unknown environments. We will refer to these opportunities as support affordances in the rest of this paper. In [2] we followed the *contacts-before-motion* paradigm [3] of dividing the generation of locomotion into two sub-steps. The first sub-step is finding a sequence of stances, which consists of contact poses between the robot and the environment. The second sub-step is generating a continuous motion linking these stances and satisfying dynamic constraints. We concentrated in [2] on solving the first sub-step using the data-driven approach of learning a probabilistic n-gram language model and training it on on human demonstrations of locomotion. Section III-A explains this approach in more detail. In [4] we combine both of these approaches and thus enabling the planning of whole-body multi-contact tasks for humanoid robots in unknown environments. This provides a link from the high-level task of perception of the environment to the lower-level task of generating a sequence of stances.

In this paper, we concentrate on linking our previous work about generating a sequence of stances to the closed-loop execution of the sequence using a QP controller. We take stance sequence generated with [2] and map it to the kinematic structure of a robot using [5]. To deal with the challenges outlined in Section III-B, we apply a post-processing step to the sequence of stances and use a closed-loop controller based on Quadratic Programming (QP) to generate joint level commands to execute the sequence. Section III-C explains our approach in detail and we evaluate the proposed approach in Section IV.

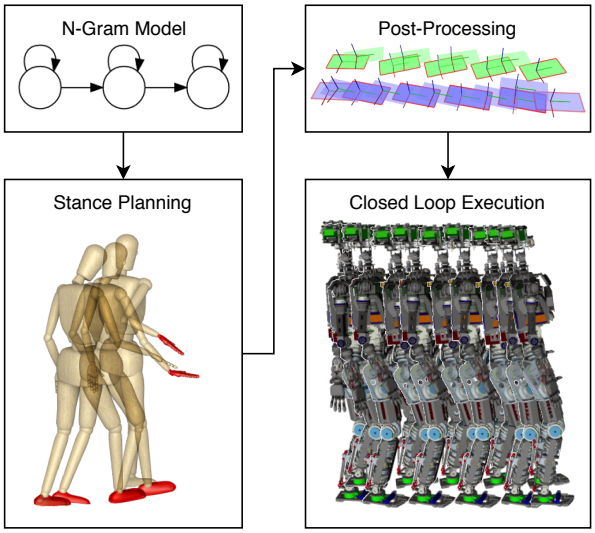


Fig. 1: We use our previous work (left) to generate a sequence of stances. In this paper (right) we post-process this sequence and execute in closed loop.

The remainder of this paper is structured as follows: Section II discusses other approaches for generating locomotion motions for robots. Section III provides details on our previous work [2] and the approach proposed in this paper. The results of our evaluation are described in Section IV and Section V concludes the paper and discusses future work.

II. Related Work

Planning motions for locomotion with multiple contacts has been addressed in the robotics literature by many authors. It is a challenging problem due to the complexity of the kinematic chains, the dynamic constraints, and the high dimensionality of the tasks. Some solutions ([6]–[8]) solve the full problem including all the dynamic equations under contact constraints, despite the involved computational cost. The second approach is the *contacts-before-motion* approach. There is extensive work on solving its first ([9]–[12]) and second ([13]–[15]) sub-task. There are several other approaches using machine learning for walking on bi- or multi-pedal robots. The approaches presented in [16] and [17] directly generate joint commands for walking. In

[1]The authors are with the High Performance Humanoid Technologies Lab, Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany, {raphael.grimm, asfour@kit.edu}
[2]Abderrahmane Kheddar is with CNRS-University of Montpellier LIRMM, Interactive Digital Human team, France. kheddar@lirmm.fr

comparison [18] and [19] use learning to optimize parameters for a pattern generator to optimize walking speed. These approaches do not perform any contact planning. In [20] the authors first learn movement primitives from optimal generated trajectories and then use these primitives to generate a close to optimal movement without the need for a computationally expensive optimization process. In comparison, this paper uses stances generated by a model trained on human demonstrations. The approach proposed in [21] uses an observed walking trajectory and post-process it for execution. The main difference in comparison to this work is, they use a whole trajectory and try to reproduce it instead of a sequence of stances.

Our controller uses Quadratic Programming (QP), which is a mathematical optimization problem and was successfully used for many robot control problems in the past ([22]–[27]). It is well suited to solve the second sub-step of the *contacts-before-motion* approach.

III. Our Approach

We generate a sequence of stances using a model trained on human demonstrations of locomotion as described in our previous work [2] and convert it to a robot's Kinematic using [5]. We apply a post-processing step to the sequence of stances and use a QP based closed-loop controller to generate joint velocities used to execute the sequence.

A. From human motion data to whole-body support poses

Beginning with human demonstrations of locomotion [28] we segment these motions according to the whole-body pose taxonomy presented in [29]. In [2], we proposed the approach of training an n-gram model for the transition probabilities between stances representing poses from our taxonomy. These stances are extracted from the segmented human motions. We utilize it to generate a sequence of multi-contact stances for bipedal robots, thus solving the first sub-step of the *contacts-before-motion* approach. The sequences we generate uses hand contacts to stabilize the motion if available. When querying for a sequence of stances we use the distance to be covered and a simplified environmental model as input. This environmental model consists of support affordances feasible to support the robot during locomotion.

A stance generated by this approach consists of joint angles for the whole robot $\mathbf{q}_S \in \mathbb{R}^n$, four booleans $c_S(lf), c_S(rf), c_S(lh), c_S(rh) \in \{1, 0\}$ indicating whether each of the four end-effectors (left/right-foot/hand) is in contact with the environment and a translation vector $t_{S_i} \in \mathbb{R}^3$ describing the relative translation from the previous stance.

$$S = \{\mathbf{q}_S, c_S(lf), c_S(rf), c_S(lh), c_S(rh), t_{S_i}\} \quad (1)$$

Using this representation, we derive further information which we use in the following steps: By accumulating the translation vectors up to a stance S_i we can determine the robot's global pose $GP^R_{S_i}$ at this stance. Using $GP^R_{S_i}$ and $\mathbf{q}_{S_i}$ the end-effector poses $GP^e_{S_i}$ ($e \in \{lf, rf, lh, rh\}$) and their orientations $\alpha_{x/y/z}$ can be calculated. These enable us to determine step height h_{S_i} for stances with a single foot support, step size s_{S_i} as the distance between two consecutive contacts of the same foot and the orthogonal distance o_{S_i} between both feet.

The stances extracted from human demonstrations are converted to a specific robot kinematic by using the Master Motor Map Framework [5].

The approaches for generating the sequence of stances and converting for a specific robot have some implications: Generating the stance sequences uses a simplified model of the environment comprised of affordances instead of meshes or occupancy grids. Hence slight collisions with the environment cannot be prevented and have to be considered in the next step. When converting the stances, differences between the robot's kinematic and a human body, i.e. the robot's feet may be much larger than a human's feet, may cause self-collisions. Furthermore, the algorithm mapping a human stance to a specific robot tries is optimization based and tries to minimize the pose error for all end-effectors. Hence, it makes a trade-off between position and rotation errors, which can lead to big rotation errors if the position is hard to reach due to the robot's kinematic limitations. Section III-B outlines further challenges encountered during execution.

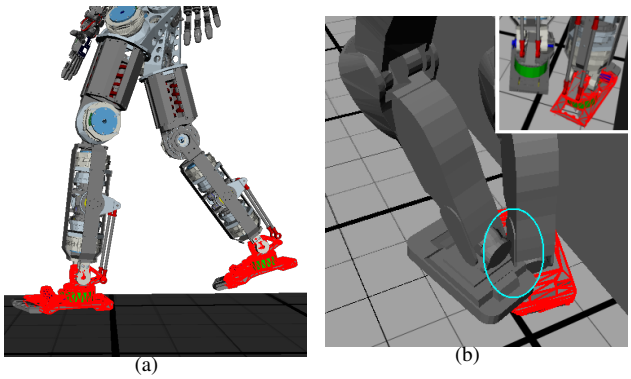
B. Challenges when executing the sequence of stances

When using a model trained on human demonstrations for stance planning, the resulting sequence of stances can contain challenges interfering with a direct execution. Such challenges include:

1) The simplified environmental model used during stance generation may result in collisions with the environment (i.e. A foot sticks into the ground).
2) The surfaces of environment and robot establishing a contact are not always parallel aligned.
3) Due to differences in the kinematic structure, the surfaces supposed to establish a contact do not always touch. (See Fig. 2(a))
4) Since the sequence of stances is generated using human demonstrations, the step size s_S can be too large for a robot.
5) For analog reasons, the orthogonal distance between both feet o_S may be too large.
6) This also can result in a too large or unnatural large step height h_S.
7) If the robot's feet are wide self-collisions between both feet or a foot and a leg may occur. (See Fig. 2(b))
8) Since the number of poses used in the n-gram model is limited, contact points between the robot and the environment can move without the separation of a contact. This leads to a jumping contact. Formal:

$$\exists\, i \in \mathbb{N}, eef \in \{lf, rf\} : \\ c_{S_i}(eef), c_{S_{i+1}}(eef), GP^{eef}_{S_i} \neq GP^{eef}_{S_{i+1}} \quad (2)$$

9) The aforementioned rotation errors of end-effectors caused by the stance conversion do not reflect the original end-effector pose, look unnatural and make



(a) (b)

Fig. 2: Feet highlighted in red are in contact with the environment. (a) shows an execution impediment where a foot in contact to the ground is not touching it. (b) shows a self-collision caused by the differences in body sizes. The big and small image shows the same stance sequence for HRP-4 and ARMAR-4 respectively. HRP-4 has a self-collision while ARMAR-4 has no self-collision.

Algorithm 1: Stance sequence post-processing

Input: Sequence of stances: S_{Seq};
Sequence Scaling Factor : scf;
Maximal step size : s_{max};
Orthogonal foot distance limits: $o_{S_{min}}, o_{S_{max}}$;
Swing foot rotation limit: $\alpha_x, \alpha_y, \alpha_z$;
Swing foot height limit: h_{min}, h_{max};

Output: Post-processed sequence of stances: S_{SeqPP}

$S_{SeqPP} \leftarrow S_{Seq}$
moveStancesUp(S_{SeqPP})
unifyContacts(S_{SeqPP})
contactFeetParallelToGroundSurface(S_{SeqPP})
contactFeetToSurfaceElevation(S_{SeqPP})
limitSwingFootRotation($S_{SeqPP}, \alpha_x, \alpha_y, \alpha_z$)
scaleTranslation(S_{SeqPP}, scf)
limitToMaxStepSize(S_{SeqPP}, s_{max})
limitOrthogonalFeetDistance($S_{SeqPP}, o_{S_{min}}, o_{S_{max}}$)
limitSwingFootHeight($S_{SeqPP}, h_{min}, h_{max}$)
return S_{SeqPP}

generating a continuous motion hard due to kinematic constraints.

Some of these challenges, i.e. too large step sizes, are amplified if the sequence of stances should be executed in a statically stable manner since the human motions used for training are only dynamically stable.

C. From whole-body support poses to continuous motion

After generating a sequence of stances using [2], we generate a continuous trajectory for the robot. We do this by first post-processing the sequence in order to mitigate the aforementioned challenges and then executing the sequence in a closed-loop QP controller generating joint velocities for each time step.

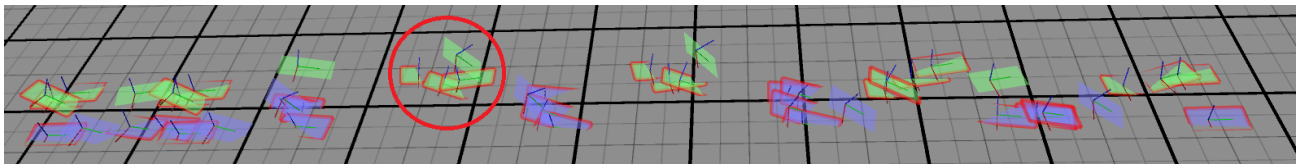
1) Post-Processing: Post-Processing is done by applying Algorithm 1 to the sequence of stances.

First, all stances are translated along the z-axis until the lower foot in contact is at the elevation for the ground support. Then we remove jumping contacts. This can be done by either selecting one of the contact poses or interpolating multiple of these poses. This paper uses the first contact pose. After this, contact poses are oriented such that they are parallel to the environmental contact surface. This is done by transforming the contact into the frame of the environmental surface and setting the contact's roll and pitch to zero. In stances with a double foot support, one foot still could be above the ground surface. Hence, all contacts are set to the z-value of the environmental surface with which they are supposed to be in contact. In combination with prior steps, this step prevents the robot from intersecting with the supporting plane. Then the swing foot's orientation is limited to prevent unnatural orientations of the foot. This is done by clamping the rotation around each axis in an interval $[+\alpha_{x/y/z}, -\alpha_{x/y/z}]$.

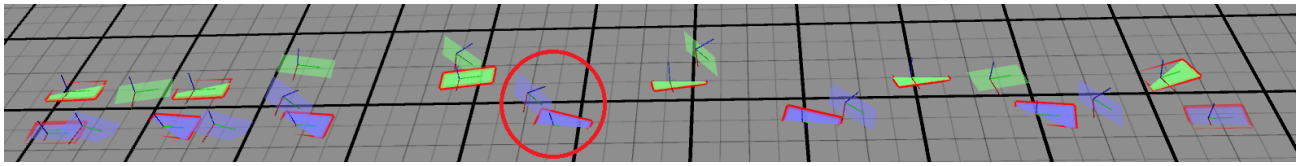
Too large step sizes are resolved using two transformations. The first transformation is scaling translations along the path the robot walks with a factor scf. The second transformation is reducing the step sizes of all steps that are too large after scaling. Only using the second transformation would suffice to make the step size feasible for the robot but would create many steps of the same length if all steps in the input sequence are too large. The first transformation is able to keep this characteristic of smaller and larger steps. It also allows us to estimate the length of the post-processed sequence. Hence we can generate a sequence of stances $S_{SeqScaled}$ of length $l_{scaled} = \frac{l}{scf}$ when we want to cover the distance l. When generating $S_{SeqScaled}$ of length l_{scaled} the availability of support affordances can be erroneous. To counter this, we also scale the availability of affordances along the path the robot has to cover. If some affordance (e.g.: hand contacts to a railing) is only available for a part of distance supposed to be covered (e.g.: for the first meter) and the scaling factor scf is 0.5, we use two meters when generating $S_{SeqScaled}$.

To prevent collisions between the robot's feet and deal with stances where the orthogonal distance between both o_S feet is too large, a minimal $o_{S_{min}}$ and maximal $o_{S_{max}}$ orthogonal distance are introduced and both feet are moved to respect these limits. This also removes any interlocking of the feet visible in a normal human gait. A more complex approach retaining this interlocking can perform self-collision checks and only move the feet further apart when they are in collision. This paper uses the first method since the implemented controller cannot handle an interlocking gait.

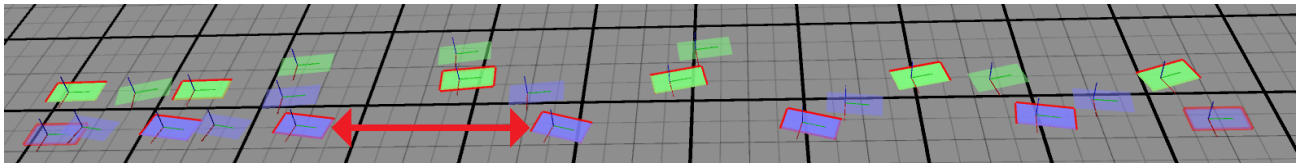
In the last transformation, the swing foot height above the ground is clamped to an interval $[h_{min}, h_{max}]$. The minimal height h_{min} prevents collisions to the ground and can be used to get a safety distance to prevent collisions in case of inaccurate execution. The maximal height h_{max} prevents unnatural high positions of the swing foot (impediment 6). Algorithm 1 and Fig. 3 show a rough outline of the post-processing as well as intermediate results.



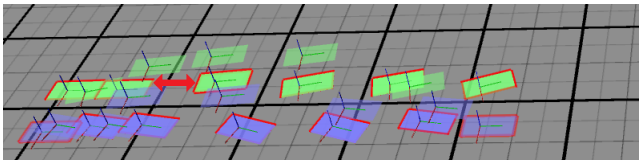
(a) after executing line 2 of Algorithm 1: The sequence contains some jumping contacts *(see red circle)*.



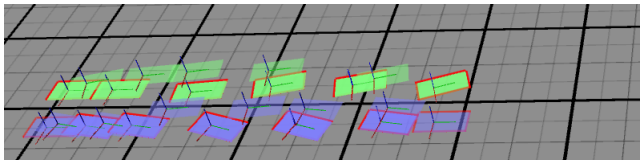
(b) after executing line 3 of Algorithm 1: Contacts now are static, but the rotation of swing and support feet are erroneous *(see red circle)*



(c) after executing line 6 of Algorithm 1: Most steps are too large to execute in a statically stable fashion. *(see red arrow)*



(d) after executing line 8 of Algorithm 1: Some steps are still larger than s_{max}. *(see red arrow)*



(e) The result of Algorithm 1

Fig. 3: Intermediate results of post-processing a sequence of stances with Algorithm 1. End-effector poses for the left (green) and right (blue) foot have a red border if they are used to form a contact to the ground. Examples for remaining execution impediments are highlighted in the images. The initial sequence is not visualized since it is completely under the floor plane.

Now all of the robot's contact surfaces are aligned to the ground, the swing foot has a sane orientation and height, all steps have a feasible size and the feet do not collide.

The joint configurations accompanying each stance are not changed. They are used as a reference configuration during execution by the controller. In our current implementation, the hands are moved by the reference configuration.

2) Closed-Loop QP controller: To generate a continuous trajectory we use a closed-loop QP controller. A QP is a mathematical optimization problem, which can be used to calculate set-points for all actuators of a robot while following a set of weighted tasks. The basic optimization problem is formulated as follows:

$$\chi^* = \text{argmin}\{\frac{1}{2} * \chi^T * Q * \chi + c^T * \chi\} \tag{3}$$

$$\texttt{subject to}: A\chi \leq b \tag{4}$$

$$c, \chi, \chi^* \in \mathbb{R}^n, Q \in \mathbb{R}^{nxn}, A \in \mathbb{R}^{mxn}, b \in \mathbb{R}^m$$

The formulation we use for the QP follows [26] and uses set-point-tasks for the position of the Center-of-Mass (CoM), joint angles and end-effector poses. For more details on the formulation we refer to [26] since explaining it in detail is beyond the focus of this paper.

Our controller is only able to execute statically stable motions on a flat surface. It uses QP for joint level control and a simple state machine for progressing through the stance sequence. The control loop is closed on joint angles and force-torque (FT) sensors in the robot's ankles. Joint angles are fed back into the QP. We use ankle forces for contact detection, which, in combination with the robot's configuration, drive progression through the state-machine. The joint velocities we calculate by solving the QP, are used as control variables, and sent to the simulator.

The contact poses are used as targets for the end-effector set-point-tasks. The joint angles accompanying each stance are used as set-point for the joint angle task. This task has a low weight and is oversteered by end-effector tasks. Hence, it primarily moves the upper body. The CoM task uses the support foot's position as the target and is only active for the x- and y-axis.

IV. Evaluation

Evaluation is done by executing post-processed stance sequences with physic simulations of ARMAR-4 and HRP-4 using the controller described in the previous section. The simulations use Bullet [30].

For evaluation we first take a look at the time required to post-process a sequence of stances depending on the distance we want to cover. Table I shows the time required is less than $150\,\mu s$ per meter if a distance of at least $5\,m$ has to be

TABLE I: Mean and coefficient of variance (CV) for the time T required for post-processing a sequence of stances to cover a requested length. The time per meter gives a more comparable measure. Post-processing was executed 1000 times for each length.

Length [m]	Mean(T) [µs]	CV(T) [µs]	Mean(T)/Length [µs/m]
0.2	153	0.19	764
1.0	244	0.17	244
5.0	674	0.11	135
10.0	1185	0.07	119
15.0	1672	0.08	111
20.0	2187	0.08	109
25.0	2681	0.09	107

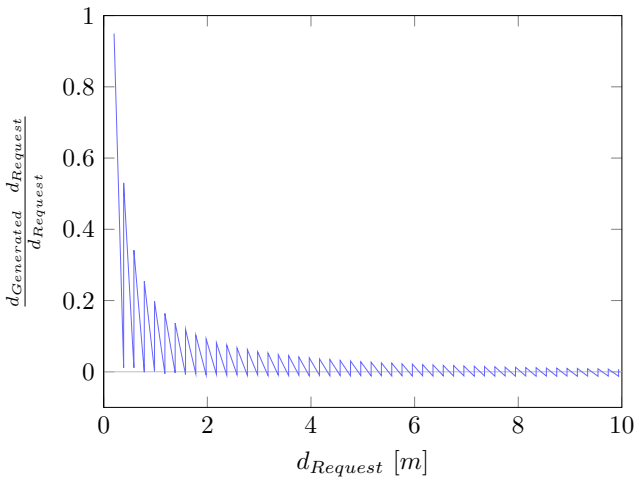


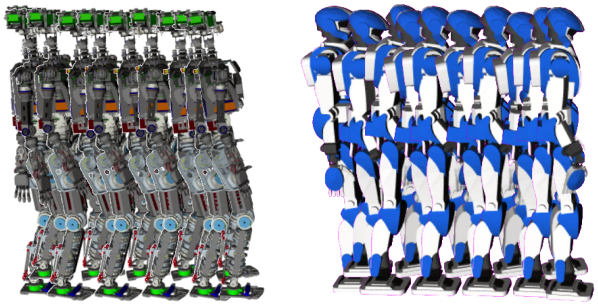
Fig. 4: Differences of the length of the post-processed stance sequence depending in proportion to requested distance to cover $d_{Request}$. When $d_{Request}$ increases the proportion decreases.

covered and post-processing requires only a few milliseconds for 25 m. Thus post-processing is fast enough for online use.

A generated stance sequence does not always cover exactly the requested distance $d_{Request}$. Furthermore post-processing can change the distance by limiting the step size s_S which is not considered when generating the sequence. Fig. 4 shows this difference as portion of $d_{Request}$. The sawtooth pattern is generated because generating a sequence of stances only adds a discrete number of steps of a certain size. If the distance to cover increases, the difference as the portion of $d_{Request}$ approaches zero.

We generated stance sequence with three different sets of available support affordances (1. Only feet, 2. Feet and left hand, 3. Feet and right hand). Since our controller does not use the hand contacts, both hands are only controlled by the set-point task for joint angles. Hence, the different sets of affordances only caused a change in the sequence of feet contacts.

Each sequence was executed ten times for each robot. Since generating the sequence of stances and post-process are deterministic, the resulting stance sequence is always identical for the same set of affordances. The controller and simulator are running asynchronously which is a slight source of non-determinism. This non-determinism only has a negligible influence and all runs of each setup were nearly identical.



(a) (b)

Fig. 5: Execution of a post-processed sequence of stances only allowing foot contacts on (a) ARMAR-4 and (b) HRP-4.

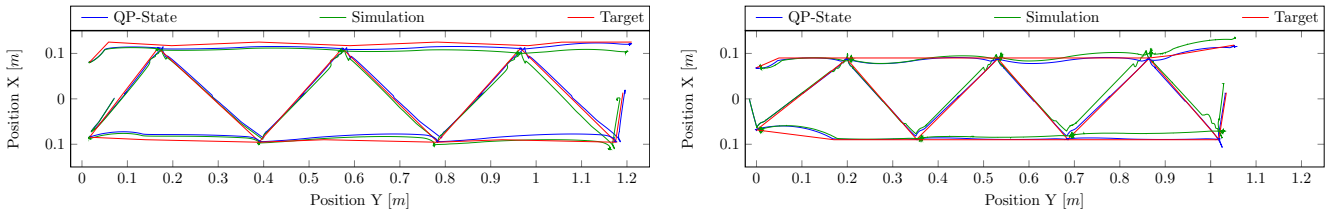
For both robots and the three sets of affordances, all executions were successful meaning the robot executed the whole sequence without toppling over or getting stuck because some step is too large. The execution of walking without any hand contacts for ARMAR-4 and HRP-4 is shown in Fig. 5. Fig. 6 shows the target (red line), the position in the simulator (green line) and the position in the controller's internal QP-State (blue) after the solver was executed for the CoM and both feet. In Fig. 6 we can see that the QP-Controller is able to follow the target. The increasing difference between the simulation and QP-State are caused by inaccuracies in the simulators calculation resulting in a drift of the robot. The inaccuracies between target and QP-State are caused by the controller's implementation. The slight differences (i.e. for the CoM) are partially caused by different set-point tasks competing with each other. Furthermore, the controller always uses the current target as set-point which allows the controller to deviate from the path as long as it reaches the target. In addition, our controller has tolerances for all targets, which allows it to transition to the next stance as long as it is close enough to the target. We do this since it speeds up execution.

V. Conclusion

In this paper, we showed how a sequence of stances containing execution impediments can be post-processed and used to generate a continuous walking motion. In the previous section, we discussed how this post-processing has to be changed to apply it to more complex scenarios. There are several areas for extending this work in the future. On the post-processing side, we want to deal with more complex scenes and include environmental obstacles as well as elevations like ramps and stairs. We want to incorporate hand contacts as discussed in the previous section. On the controller side, we want to execute the motion on a real robot in a dynamically stable fashion.

Acknowledgments

We would like to thank Stéphane Caron, Vincent Samy, Arnaud Tanguy, Kevin Chappellet and Pierre Gergondet from LIRMM (Montpellier Laboratory of Informatics, Robotics and Microelectronics) for providing their QP control framework and supporting us in all questions related to QP control.



(a) (b)

Fig. 6: Execution of a post-processed sequence of stances only allowing foot contacts on (a) ARMAR-4 and (b) HRP-4. Both robots execute the same sequence. The distance to cover is one meter for HRP-4. The motion is in direction of the y-axis. The top line and bottom lines are respectively for the left and right foot while the middle lines show the CoM position.

REFERENCES

[1] P. Kaiser, E. E. Aksoy, M. Grotz, and T. Asfour, "Towards a hierarchy of loco-manipulation affordances," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Deajeon, Korea, 2016, pp. 2839–2846.

[2] C. Mandery, J. Borràs, M. Jöchner, and T. Asfour, "Using language models to generate whole-body multi-contact motions," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea, October 2016, pp. 5411–5418.

[3] K. Bouyarmane and A. Kheddar, "Humanoid robot locomotion and manipulation step planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.

[4] P. Kaiser, C. Mandery, A. Boltres, and T. Asfour, "Affordance-based multi-contact whole-body pose sequence planning for humanoid robots in unknown environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3114–3121.

[5] O. Terlemez, S. Ulbrich, C. Mandery, M. Do, N. Vahrenkamp, and T. Asfour, "Master motor map (mmm) - framework and toolkit for capturing, representing, and reproducing human motion on humanoid robots," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, Madrid, Spain, November 2014, pp. 894–901.

[6] L. Sentis and M. Slovich, "Motion planning of extreme locomotion maneuvers using multi-contact dynamics and numerical integration," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, Oct 2011, pp. 760–767.

[7] S. Lengagne, J. Vaillant, E. Yoshida, and A. Kheddar, "Generation of whole-body optimal dynamic multi-contact motions," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1104–1119, 2013.

[8] L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Souères, and J. Y. Fourquet, "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 346–362, April 2013.

[9] J. Chestnutt, M. Lau, G. Cheung, J. Kuffner, J. Hodgins, and T. Kanade, "Footstep planning for the honda asimo humanoid," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005, pp. 629–634.

[10] K. Bouyarmane and A. Kheddar, "Multi-contact stances planning for multiple agents," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 5246–5253.

[11] ——, "Static multi-contact inverse problem for multiple humanoid robots and manipulated objects," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, Dec 2010, pp. 8–13.

[12] K. Hauser, T. Bretl, and J. C. Latombe, "Non-gaited humanoid locomotion planning," in *5th IEEE-RAS International Conference on Humanoid Robots, 2005.*, Dec 2005, pp. 7–12.

[13] S. Feng, X. Xinjilefu, C. G. Atkeson, and J. Kim, "Optimization based controller design and implementation for the atlas robot in the darpa robotics challenge finals," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, Nov 2015, pp. 1028–1035.

[14] S. Carón and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3d com accelerations," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, Nov 2016, pp. 550–557.

[15] S. Lohmeier, T. Buschmann, and H. Ulbrich, "System design and control of anthropomorphic walking robot lola," *IEEE/ASME Transactions on Mechatronics*, vol. 14, no. 6, pp. 658–666, Dec 2009.

[16] H. Benbrahim and J. A. Franklin, "Biped dynamic walking using reinforcement learning," *Robotics and Autonomous Systems*, vol. 22, no. 3, pp. 283–302, Dec. 1997.

[17] X. Xiong, F. Wörgötter, and P. Manoonpong, "Adaptive and Energy Efficient Walking in a Hexapod Robot Under Neuromechanical Control and Sensorimotor Learning," *IEEE Transactions on Cybernetics*, vol. 46, no. 11, pp. 2521–2534, Nov. 2016.

[18] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *2004 IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04*, vol. 3, Apr. 2004, pp. 2619–2624 Vol.3.

[19] D. J. Christensen, J. C. Larsen, and K. Stoy, "Fault-tolerant gait learning and morphology optimization of a polymorphic walking robot," *Evolving Systems*, vol. 5, no. 1, pp. 21–32, Mar. 2014.

[20] D. Clever, M. Harant, K. Mombaur, M. Naveau, O. Stasse, and D. Endres, "COCoMoPL: A Novel Approach for Humanoid Walking Generation Combining Optimal Control, Movement Primitives and Learning and its Transfer to the Real Robot HRP-2," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 977–984, Apr. 2017.

[21] J. Morimoto, J. Nakanishi, G. Endo, G. Cheng, C. G. Atkeson, and G. Zeglin, "Poincar #233;-Map-Based Reinforcement Learning For Biped Walking," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Apr. 2005, pp. 2381–2386.

[22] Y. Abe, M. Da Silva, and J. Popović, "Multiobjective control with frictional contacts," in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*. Eurographics Association, 2007, pp. 249–258.

[23] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-based locomotion controllers," in *ACM SIGGRAPH 2010 Papers*, ser. SIGGRAPH '10. New York, NY, USA: ACM, 2010, pp. 131:1–131:10.

[24] K. Bouyarmane and A. Kheddar, "Using a multi-objective controller to synthesize simulated humanoid robot motion with changing contact configurations," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2011, pp. 4414–4419.

[25] K. Bouyarmane, J. Vaillant, F. Keith, and A. Kheddar, "Exploring humanoid robots locomotion capabilities in virtual disaster response scenarios," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, Nov 2012, pp. 337–342.

[26] J. Vaillant, A. Kheddar, H. Audren, F. Keith, S. Brossette, K. Kaneko, M. Morisawa, E. Yoshida, and F. Kanehiro, "Vertical ladder climbing by the hrp-2 humanoid robot," in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov 2014, pp. 671–676.

[27] J. Vaillant, K. Bouyarmane, and A. Kheddar, "Multi-character physical and behavioral interactions controller," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 6, pp. 1650–1662, June 2017.

[28] C. Mandery, O. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour, "The KIT whole-body human motion database," in *International Conference on Advanced Robotics (ICAR)*, Istanbul, Turkey, July 2015, pp. 329–336.

[29] J. Borràs and T. Asfour, "A whole-body pose taxonomy for loco-manipulation tasks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, October 2015, pp. 1578–1585.

[30] Erwin Coumans, "Bullet physics engine," http://bulletphysics.org, 2010.