Path planning with force-based foothold adaptation and virtual model control for torque controlled quadruped robots

Alexander Winkler[†,⋆] Ioannis Havoutis[†] Stephane Bazeille[†] Jesus Ortiz[†] Michele Focchi[†]
Rüdiger Dillmann[⋆] Darwin Caldwell[†] Claudio Semini[†]
[†]Department of Advanced Robotics, Istituto Italiano di Tecnologia, via Morego, 30, 16163 Genova, Italy.
[⋆]Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany.

***Abstract*— We present a framework for quadrupedal locomotion over highly challenging terrain where the choice of appropriate footholds is crucial for the success of the behaviour. We use a path planning approach which shares many similarities with the results of the DARPA Learning Locomotion challenge and extend it to allow more flexibility and increased robustness. During execution we incorporate an on-line force-based foothold adaptation mechanism that updates the planned motion according to the perceived state of the environment. This way we exploit the active compliance of our system to smoothly interact with the environment, even when this is inaccurately perceived or dynamically changing, and update the planned path on-the-fly. In tandem we use a virtual model controller that provides the feed-forward torques that allow increased accuracy together with highly compliant behaviour on an otherwise naturally very stiff robotic system. We leverage the full set of benefits that a high performance torque controlled quadruped robot can provide and demonstrate the flexibility and robustness of our approach on a set of experimental trials of increasing difficulty.**

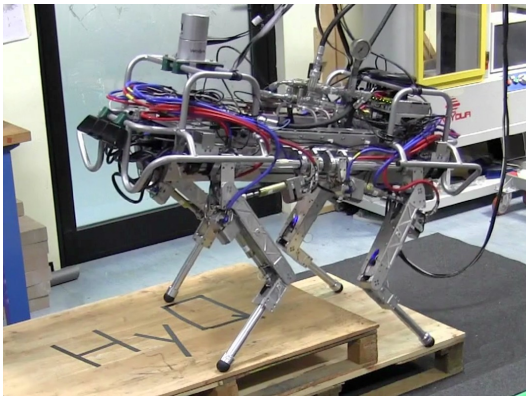

Fig. 1. The hydraulically actuated and fully torque controlled quadruped robot HyQ. Here shown during an experimental trial where it steps onto a structure that consists of two 0.15m high pallets.

I. Introduction

The ability to move from one place to another is one of the most basic and important skills in nature. Although wheels offer great efficiency they are not suited for crossing highly unstructured and challenging terrain. Nature shows that legged locomotion has far more potential in these areas in terms of agility and performance.

This paper presents the newest development in a stream of research that aims to increase the autonomy and flexibility of legged robots in unstructured and irregular environments. We present a framework for quadrupedal locomotion over highly challenging terrain where the choice of appropriate footholds by the robot is crucial for the success of the behaviour. In addition, our approach combines a *virtual model* based controller that guarantees the overall compliant behaviour of the system while also maintaining a high level of accuracy in trajectory execution.

The evaluation of our approach is done on the hydraulically actuated quadruped robot HyQ [1], shown in Fig. 1. With the framework presented in this paper, HyQ is able to cope with unperceived obstacles, traverse highly irregular terrain, walk over 15 cm pallets and climb stairs. This performance is achieved by global evaluation of the terrain, planning of appropriate footholds and robust execution of these steps. In addition, force-based feedback is used to detect early or no contact of the swing-leg and update the planned motion on-the-fly. This allows us to execute the planned motions even in inaccurately perceived or dynamically changing environments.

We present experimental trials on a set of environments with increasing difficulty. We use real maps of the environment that are created offline. The environment is changed during trials to underline that our framework is robust to such dynamic changes. We experiment on flat ground with unperceived obstacles, on crossing a gap with distinct stepping stones where we purposely remove one of the stepping stones, and on climbing one and two pallets. Note that the height of one pallet is 15 cm, that is 20% of the leg length of HyQ fully stretched.

Our contributions include online replanning of motions based on sensed contact forces, a virtual model controller for a fully torque controlled quadruped robot with on-board state estimation for accurate but compliant robot behaviour, and on-the-fly adaptation of swing-leg trajectories.

The rest of the paper is structured as follows. In Section II we discuss related work on robotic legged locomotion. Section III describes the different steps that generate the initial plan for the robot to follow. Section IV describes the adaptive execution of the plan, the local plan updating based on force-feedback, the estimated robot state, and the virtual model controller employed by our approach. Section V evaluates our approach in real world experimental trials. In Section VI we conclude and present ideas for future work.

email: alexander.winkler@aol.com, ruediger.dillmann@kit.edu, {ioannis.havoutis, stephane.bazeille, jesus.ortiz, michele.focchi, darwin.caldwell, claudio.semini}@iit.it.

II. Related Work

Statically stable walking, keeping the robots's center of gravity (CoG) inside the polygon formed by its supporting feet, was first identified by Muybridge [2] and mathematically evaluated by McGhee and Frank [3]. This work was extended to facilitate walking over irregular terrain [4].

In environments where smooth, continuous support is available (flats, fields, roads, etc.), where exact foot placement is not crucial for the success of the behaviour, legged systems can utilize a variety of more dynamic gaits, e.g. trotting, galloping. Marc Raibert studied the principles of locomotion and dynamic balancing with legged robots [5]. The quadruped *BigDog* and *LS3* are a recent extension of his work. While *BigDog* is able to traverse irregular terrain using a reactive controller, the footholds are not planned in advance. Similar performance can be seen on HyQ, that is able to overcome obstacles through reactive control [6], [7] or reflexes [8]. For more complex environments, with obstacles like large gaps or stairs, such systems quickly reach their limits as a higher level movement planning process that takes the environment into account is required.

In terrain with only a few possible discrete footholds, e.g. steps, stairs, cluttered rooms, legged robots can employ a range of typically non-gaited static or quasi-static locomotion strategies that rely more on accurate foothold planning, and consequentially on features of the terrain. The DARPA Learning Locomotion Challenge excelled the development of footstep planning over rough terrain. It resulted in a number of successful control architectures [9], [10], [11], [12], [13], [14] to plan and execute footsteps to traverse challenging terrain. Rebula et al. [9] avoids global footstep planning by simply choosing the next best reachable foothold. This can cause the robot to locally navigate into an insurmountable obstacle. To avoid this, some controllers [15], [13] *globally* plan the complete footsteps from start to goal, though in this case time consuming replanning is necessary in case of slippage or deviation from the planned path. The approach in [12] stands between the two above mentioned methods and plans a global rough body path to avoid local minima, but the specific footholds are chosen only a few steps in advance. This reduces the necessary time for replanning in case of slippage, while still considering a locally optimal plan. Pongas et al. [10] focused mainly on generating a smooth CoG trajectory independent of the foothold pattern. Previously we have experimented with coupling on-board perception with a trotting controller and a simplistic crawl gait controller [16].

Our approach has many similarities to the above mentioned work by Kolter et al. [11] and Rebula et al. [9]. The latter has the most in common with our approach. At this point we also wish to highlight that we do not make use of any external state measuring system, e.g. a Vicon marker-based tracker system commonly used in many of the aforementioned approaches.

III. Initial Planning

The complex task of planning and executing a path from a start to a goal position is divided into a set of different solvable sub tasks (Fig. 2). A *stance* $\mathcal{F}$ is a tuple

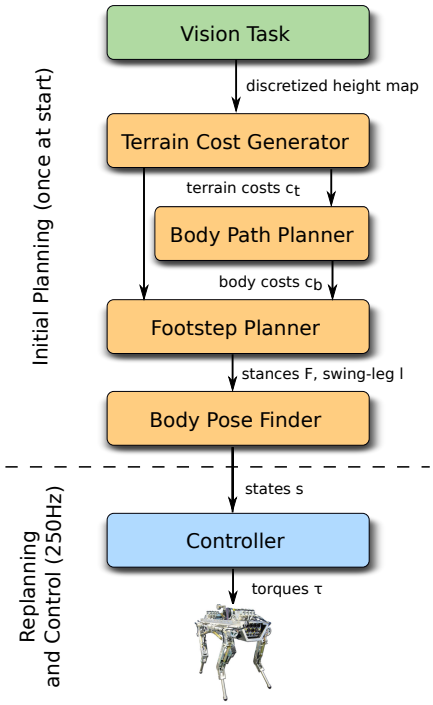

Fig. 2. An overview of the entire framework for quadrupedal path planning and execution over rough and irregular terrain.

$(f_{lf}, f_{rf}, f_{lh}, f_{rh})$ of four feet locations $f \in \mathbb{R}^3$ in the world frame – left-front, right-front, left-hind, right-hind respectively. A *state* s includes a stance $\mathcal{F}$, the position and orientation of the robot body X_b in the world frame and the swing-leg[1] l as

$$s = \{\mathcal{F}, X_b, l\}. \tag{1}$$

The goal of the initial planning is to find a sequence S of robot states s from start to goal as

$$S = (s_i)_{i=0}^{n} = (s_0, s_1, \ldots, s_n), \tag{2}$$

where $s_i \in S$ describes the ith state in the sequence and n the total number of steps necessary to cross the terrain. This sequence S is then reactively executed by the controller described in Section IV. The different modules of the initial planning architecture are described in the following subsections.

A. Vision Task

Accurate perception of the environment is important for path planning. We use a real map of the environment that is generated offline using a *Microsoft Kinect* sensor and the *KinectFusion* [17] algorithm (Fig. 3). The point cloud is converted to a discretized height map (cell resolution: 1 cm) and given to the terrain cost generation module. At this point *online* perception and map building is part of ongoing work that comes close to our line of research, but is beyond the scope of this paper.

[1]The swing-leg is a number between 1 and 4 referring to the leg to swing to reach state s.

B. Terrain Cost Generator

As in [11] the terrain cost generator creates a *terrain cost map* from the height map, where the terrain cost c_t reflects how desirable it is to place a foot at a specific location. To create this terrain cost map we choose the following characteristics, also called *features*, that affect the quality of a foothold:

1) standard deviation of the heights around the cell
2) highest cell in an area in front of the evaluated cell
3) highest cell in an area behind the evaluated cell
4) estimated slope through regression
5) estimated curvature through regression

These 5 features are calculated on 3 different scales as done in [11], [19] to generate a cost vector $\mathbf{c}_c \in \mathbb{R}^{15}$ for each cell. The individual costs in this cost vector $\mathbf{c}_c$ are weighed by a weight vector[2] $\mathbf{w}_1 \in \mathbb{R}^{15}$ to produce the terrain cost $c_t = \mathbf{w}_1^T \mathbf{c}_c$ for each cell. A high terrain cost c_t implies that this is not a good foothold to choose, based on the terrain characteristics.

C. Body Path Planner

Next we use the terrain cost map to calculate a *body cost map* that reflects how desirable it is for the CoG of the robot to be at that position. The cheapest path through this body cost map leads the robot through *regions* of good footholds. This reduces the search area for the foothold selection and decreases the computational load.

The body cost for each cell c_b is calculated from

$$c_b = w_{21}\bar{c}_t + w_{22}c_{goal} + w_{23}\hat{c}_{roll,pitch}. \quad (3)$$

with the weight factors[2] $\mathbf{w}_2$. The first cost value $\bar{c}_t$ is the previously calculated terrain cost averaged over an area around each of the four footholds in nominal stance[3] [11]. In our approach we added a goal deviation cost c_{goal}, which gives a low cost to cells that are closer to the direct line from start to goal. A large weight factor w_{22} will cause the robot to walk in a straight line towards the goal, paying no attention to possibly difficult regions of the terrain. The second difference to previous work is the estimated roll and pitch cost $\hat{c}_{roll,pitch}$. It estimates the average terrain height in an area below the left feet compared to the right feet. A big difference will most likely cause the robot to have a large roll angle to compensate for this difference, which decreases the stability of the stance. The same estimation is done for the pitch using the average terrain height below the front and hind legs. Finally we use Dijkstra's search algorithm [20] to find the path with the lowest total body costs c_b through this map. We can use Dijkstra here since we are not limited by the time taken by the search, i.e. the robot is allowed to plan the path as long as it takes before executing the first step. In practice more efficient search approaches such as *A**, *D**, *D*lite*, etc, can be used.

[2]All weight factors $\mathbf{w}$ were experimentally tuned and slightly adjusted according to the desired robot behaviour and terrain.

[3]The nominal stance is a fixed joint configuration that creates a "natural" stance of the robot. Fig. 4 shows a simplified representation of the robot in nominal stance (grey).

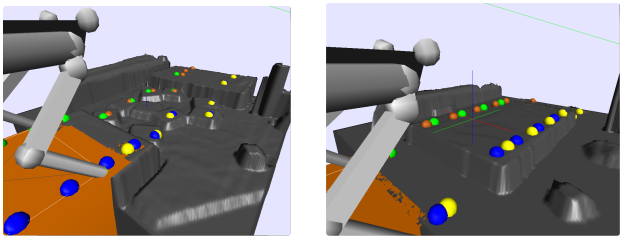

Fig. 3. Maps generated by a the KinectFusion algorithm displayed in the simulation environment SL [18], highlighting the planned footholds. *Left*: the map of the gap with the stepping stones, *right*: the map with the single pallet.

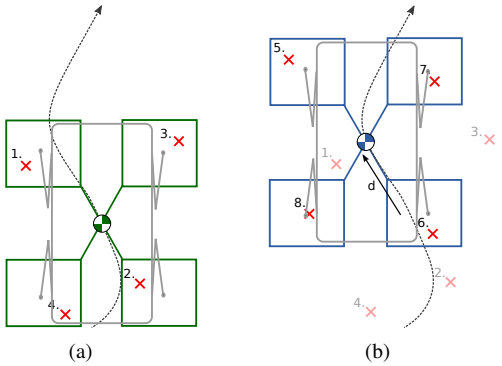

Fig. 4. The robot in nominal stance (grey) with the foothold search area (green and blue squares) for each leg. The gray dotted line shows the body path as previously planned. (a) The best foothold (red x) around each nominal footholds is chosen in order of the gait sequence; (b) the virtual CoG is moved by a step-length d and the next 4 best footholds inside the blue search areas are chosen.

D. Footstep Planner

The footstep planner searches for *specific* footholds that roughly guide the robot along the previously planned body path (see Fig. 4). As described by [11] the 4 nominal footholds corresponding to a virtual CoG position on the planned body path are calculated. In each area around these nominal footholds the foothold with the lowest foothold cost c_f is chosen. After the best 4 footholds are chosen in a specific swing-leg sequence, the virtual CoG is moved by the step-length d and the procedure is repeated.

The foothold cost c_f used to choose a specific foothold is calculated from

$$c_f = w_{31}c_t + w_{32}c_{bias} + w_{33}c_{supp} + w_{34}c_{roll,pitch}, \quad (4)$$

with the weight factors[2] $\mathbf{w}_3$. The terrain cost c_t was previously calculated by the terrain cost generator. The bias cost c_{bias} ensures that the robot chooses footholds based on the planned body path by assigning the lowest cost to the center of each search area.

We built on this approach by including a support triangle cost c_{supp} which is based on the size of the support triangle[4] if this foothold is chosen. A larger support triangle leaves more flexibility to position the CoG in a statically stable way. We also added a roll and pitch cost $c_{roll,pitch}$ to the foothold cost. Compared to the *estimated* roll and pitch cost

[4]The triangle formed by the location of the stance feet. For static stability the CoG must always be inside this triangle.

$\hat{c}_{roll,pitch}$ in the body path finder, this cost knows the *precise* location of the chosen footholds and calculates the height difference and score based on these. Our additional costs greatly improve the quality of the chosen footholds, since they "plan ahead" according to which stances $\mathcal{F}$ are statically executable. Stances that are difficult to execute are eliminated early in the planning process, which reduces the complexity to find a suitable body position and orientation later on.

Our approach also allows deviation from a specific swing-leg sequence to allow more flexibility in foot placement and greatly increase locomotion speed. When attempting a long step with a front leg, it is better to first move the hind legs close to the front legs. This avoids the hind legs being overextended while swinging the front leg. Another reason to change the swing-leg sequence is to always use and create the biggest possible support triangles. After the foothold with the lowest foothold cost c_f is determined, our planner checks the swing distance to this foothold. If this distance is smaller than a threshold (e.g. 5 cm), moving this foot has little or no advantage for the robot in relationship to the effort and time the step takes. In this case the planner skips this step and the next leg in the cycle is evaluated.

E. Body Pose Finder

As mentioned earlier we are searching for a sequence S of states s from start to goal. At this point a sequence of stances $\mathcal{F}$ and the corresponding swing-leg l have been found. The body pose finder creates intermediate states for static stability and plans the optimal body pose X_b for each state.

An intermediate state $s_{i\to(i+1)}$ is inserted between every state in S as

$$S = (s_i)_{i=0}^{n} = (s_0, s_{0\to1}, s_1, s_{1\to2}, \ldots, s_n), \qquad (5)$$

that moves the CoG into the current support triangle so the robot is statically stable when executing the next step. The CoG is moved between the center of the support triangle and the average position of the feet in the next stance. Moving the CoG towards the next stance limits backward motion of the CoG. The planner checks if the calculated CoG is inside the support triangle, reduced by a safety margin to account for inaccuracies in execution, and re-plans more conservatively if necessary. Similar to [11] we choose the vertical position of the body based on the height of the feet raised by a desired body height. The pitch of the body is based on the height difference between the front and the hind feet. We improve on this concept by adjusting the roll depending on the height difference between the left and the right feet.

IV. Path updating, Re-Planning and Control

After the initial plan has been generated, the sequence of states is executed. In our approach we combine the initial plan with a number of online elements that are crucial to the success of the behaviour. An open-loop execution of the initial plan is doomed to failure due to a number of factors, such as the robot being (desirably) compliant, incomplete or inaccurate knowledge of the environment and foot slippage. Though in short movements such nuisances can be neglected, our aim is to use our platform in large scale environments where errors from the aforementioned

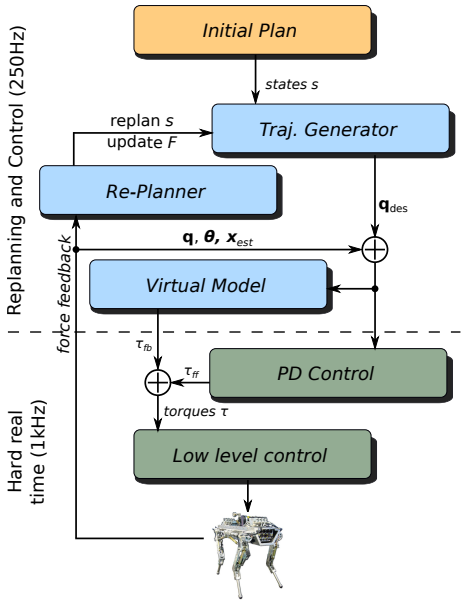

Fig. 5. Detailed view of the controller that executes the initial plan. The controller compliantly executes, updates and re-plans the initial plan on-the-fly.

factors quickly accumulate. Our approach combines online feet trajectory generation, a force-based foothold adaptation method and a virtual model controller that, alongside the very-low gain PD controller at the joint level, keep the robot highly compliant and the overall behaviour highly accurate.

A. Trajectory Generator

The trajectory generator interpolates the states in S at a rate of 250 Hz and uses inverse kinematics to map these interpolated Cartesian states to desired joint states $\mathbf{q}_{des} \in \mathbb{R}^{12}$. While the position and orientation of the body X_b are simply interpolated between two states s, the swing-leg must be "consciously" raised and lowered. Based on the current position of the leg and the desired position, an optimal swing-leg trajectory is generated online. For steps on flat ground the height of the swing is low to increase accuracy and speed of execution. If the desired foothold lies higher than 10% of the leg length, a trajectory is generated that swings the leg outward. This motion allows the robot to step onto obstacles that it would otherwise collide with (*stumble*) due to joint range limitations while swinging up. Adjusting the roll and pitch of the robot *while* swinging a leg creates whole body motions that greatly increase the kinematic reachability. This allows us to execute swing-leg trajectories and stances that are otherwise not possible (see Fig. 8). The time to execute these trajectories depends on the travel distance. Short motions are assigned less time to achieve optimal performance.

B. Force-based Foothold Adaptation and Plan Update

The execution of a motion can never be absolutely accurate and the environment can never be perfectly known. Because of this the robot may collide with the environment before or after the expected contact. Although the active compliance limits the effect of such unexpected contacts and ensures

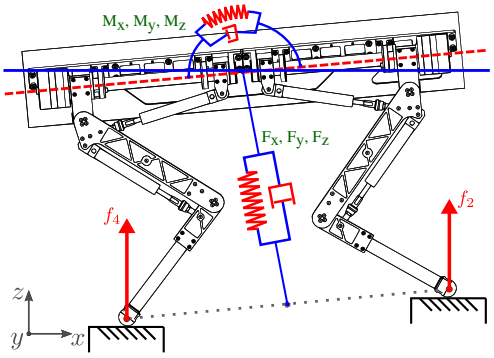

Fig. 6. The virtual elements used to calculate forces and moments around the trunk of the robot. The virtual forces and moments are transformed to forces at the feet and subsequently to feedforward torques for the legs that are in stance.

a smooth interaction with the environment, the robot is still not aware that his foot is now higher or lower than expected. This is likely to cause the robot to fail to execute its desired motion in future steps.

To avoid such difficulties, online *force feedback* detects the contact condition of the swing-leg. This is done by computing the force at the foot, using the load and torque sensors of each of the robot's joints and the leg Jacobian of the current leg state. If early contact is detected the swing is stopped and the subsequent states are re-planned so that the *haptically* sensed difference is reflected in the robot's perception of the environment. The controller re-plans the optimal CoG positions, orientations and swing-leg trajectories on the fly. This enables the robot to continue the initial sequence in spite of unperceived obstacles up to 20% of its leg length. In the opposite case, if no forces are sensed after the swing has been completed, then the target foothold is lowered by 2 cm and the swing time is prolonged until the robot "feels" the ground. This haptic force feedback makes our approach very robust to perception inaccuracies and contributes greatly to the overall stability of the locomotion behaviour.

C. Virtual Model

As outlined before, we aim for a highly compliant behaviour of our system in order to naturally cope with the environment and estimation inaccuracies. Nonetheless we require very precise foothold landing, something crucial for the overall success for the behaviour. We follow a virtual model control approach similar to [21]. We calculate virtual forces (F_x, F_y, F_z) and moments (M_x, M_y, M_z) according to a desired state and the current state of the system (Fig. 6). The current state of the system is computed with the approach of [22], without the use of any external sensing (e.g. a VICON motion capture system). The desired state on the other hand is planned and re-planned/updated according to the current situation, e.g. force-based foothold adaptation. The virtual forces and moments are then transformed to forces f_i that the feet in contact need to apply. Theses forces can be optimized over with a number of methods. In our case we use a least squares optimization that provides the least norm solution, i.e. the minimum force solution. The feet forces are subsequently mapped to feedforward torques τ_{ff} (Fig. 7) for the joint actuators of the legs that are in stance, using the Jacobian of the system's current state. Formally this is done by

$$\tau_{ff} = J^T \mathbf{f}, \tag{6}$$

where $\mathbf{f}$ is the vector of (linear) forces that each foot in stance needs to apply to emulate the virtual model behaviour, and J is the Jacobian of the legs that are in contact. We wish to underline here that without the virtual model controller, successful execution of the planned motions is impossible.

D. PD Controller

We use a PD controller with very low feedback gains for all joints of the robot, as we are aiming at a very compliant behaviour, something important for smooth interaction with the environment. This is the joint level controller that runs on a 1 kHz control loop (Fig. 5). Note that the leg in swing phase is controlled only through this loop as the virtual model produces torque inputs only for the legs in stance. The feedback torque τ_{fb} that the PD control loop provides is generally very small in comparison to the feed-forward control input that comes from the virtual model. Fig. 7 is an illustrative example of this hybrid control setup, where the virtual model torque accounts for most of the control input.

V. Experimental Results

The following section describes the experiments conducted to validate the performance of our controller and the obtained results.

A. Experimental Setup

For each of our experiments, we only give the robot the x,y-coordinates of the goal in front of it. We validate the performance of our framework in 4 different scenarios as seen in Fig. 8. In the first experiment the robot plans a path for flat, obstacle-free terrain. Unperceived by the robot, random obstacles are then placed in the robot's path. The second experiment consist of two pallets connected by a sparse path of stepping stones. The pallets are 1.2 m apart and the stepping stones lie 0.08 m lower than the pallets.

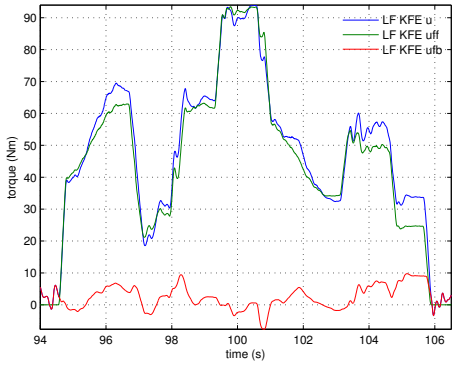

Fig. 7. Torque profiles of one knee joint during a complete gait cycle. The leg swings until 94.5 s and remains in stance while the other 3 legs swing until 106 s. In red the on average very low torque input that the PD controller produces [*ufb*]. In green the torque input that the virtual model controller produces [*uff*]. This accounts for most of the torque that the joint produces throughout locomotion. The blue signal [*u*] is the sum of the two aforementioned terms and the torque the low-level controller aims to track.

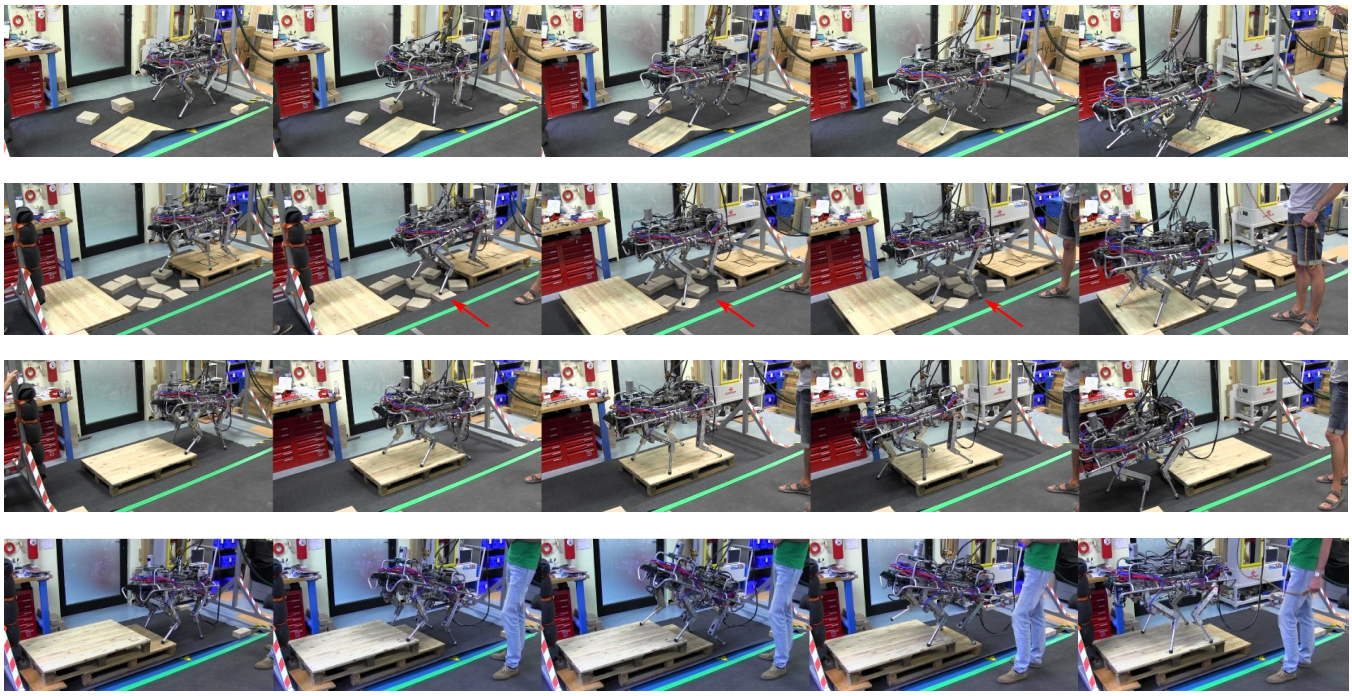

Fig. 8. Snapshots of the 4 experimental trials used to evaluate the performance of our framework. From top to bottom: flat ground with random, unperceived obstacles; crossing over stepping stones while the highlighted stone was removed during the trial; climbing over a 15 cm tall pallet; climbing a stair-like structure consisting of two stacked pallets.

Unperceived by the robot, a set of two stepping stones is removed after the front legs have used this foothold. In the third experiment the robot must climb over a pallet with dimensions 1.2 m $\times$ 0.8 m $\times$ 0.15 m. In the last experiment the robot must climb up two stacked pallets. The height of each pallet is 0.15 m (20% of the leg length) and the offset of the pallets is 0.4 m.

B. Results and Discussion

The first experiment shows the ability of our controller to overcome unperceived obstacles. If the force feedback senses unexpected contact with an obstacle the swing-leg sequence is terminated. The force feedback senses the wood and the stone and terminates the swing-leg sequence. Following this, the orientation of the body is replanned to maximise kinematic reachability. Although our approach is robust to adapting to unknown terrain, scenarios where the foot initially touches down but slips after a few seconds are difficult to detect, since the force feedback is used only for detecting contact of the swing-leg and does not affect the legs in stance (see Table I).

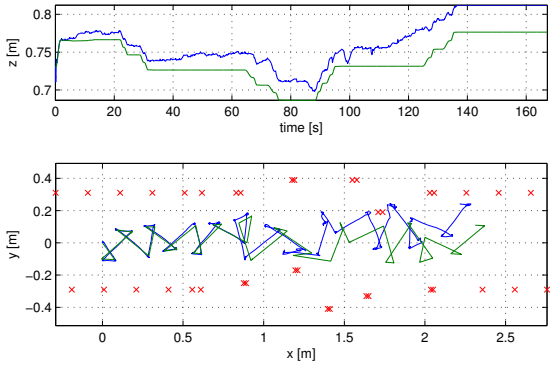

Fig. 9. The estimated absolute position (global) of the robot body with respect to its starting position while crossing the stepping stones trial environment. *Top*: the height of the robot's body. *Bottom*: the top-down view of the robot's position. Red crosses represent the chosen footholds, green lines represent the planned path (reference) and blue lines represent the state estimate that combines the IMU with the legged odometry. Note that the estimated state may drift.

In the second experiment our controller demonstrates the ability to overcome irregular terrain. The deviation from the standard swing-leg sequence is crucial to traverse these stepping stones. Additionally, we remove one stepping stone during the experiment, which causes the robot to slowly "feel" for contact moving the foot down. Even after deviating 8 cm from the planned foothold, the robot is able to locally re-plan its motion and successfully cross the terrain. While our framework is able to adapt to complete misses in footholds, a contact followed by a slip or a tipping stepping stone has proven difficult to handle. The performance of the on-board state estimation for this trial can be seen in Fig.9.

In the third experiment whole body motions are crucial to

TABLE I
RESULTS AVERAGED OVER 10 TRIALS PER SETUP

Terrain	Success rate	Avg. Speed (cm/s)
Flat Ground	80%	1.83
Stepping Stones	70%	1.70
Pallet	90%	2.11
Two Pallets	80%	1.76

overcoming the pallet. Motions like crouching down before stepping off the pallet are necessary to increase the kinematic reachability and succeed in crossing this terrain. Due to the joint limits it is difficult to raise the leg near vertically when stepping onto the pallet. The adaptive swing-leg trajectory, causing the outward swing motion, greatly improved the performance of our system (see Table I).

The fourth experiment, similar to the previous one, shows the robot at even more pronounced pitch angles. The increased difficulty in this experiment comes from the fact that the robot needs to simultaneously step on the ground, the first step and the top pallet in order to successful complete the trial. Again, whole body motions are key, as the reach of the robot needs to be maximized. Failures to cross the terrain in this and the previous setup are due to the insufficient joint torques the robot can supply in extreme joint configurations, due to the diminishing lever arm close to joint limits.

The success rates of the aforementioned experimental trials can be seen in Table . I and provide substantial evidence of the robustness of our path planning and control framework. Additionally, the reader is strongly encouraged to view the accompanying video as it provides the most intuitive way to demonstrate the performance of our framework.

VI. Conclusion

We presented a framework for quadrupedal locomotion over highly challenging terrain where the choice of appropriate footholds is crucial for the success of the behaviour. We showed how a body path and the footholds are planned and how the planned motions are robustly executed. To do so, we explained the benefits of a force-based foothold adaptation method and the subsequent plan update. We presented the use of a virtual model controller that ensures the overall compliance of the system when interacting with the environment, while also providing highly accurate motion execution. We evaluated our approach on a number of experimental trials of increasing difficulty with irregular, structured and unstructured terrains. Our framework has proven effective in surpassing highly challenging terrains and robust to inaccurately perceived and dynamically changing environments.

In the future we aim to extend our planning and control framework to handle dynamically planned motions, where the planner will rely on dynamic models of the robot, e.g. table-cart or inverted pendulum, to produce more natural and animal like motions. We also aim to extend our virtual model controller to include more information in the optimization step, such as the torque limits/capabilities of the joint actuators and information about the feet contacts, e.g. surface norms and estimated friction coefficients. In addition, another part of our group is currently working on on-line map building and localization, something that will allow our framework to autonomously locomote past any traversable terrain.

Acknowledgements

This research is funded by the Fondazione Istituto Italiano di Tecnologia. The authors would like to thank the scholarship program *interACT* and Prof. Georg Bretthauer from KIT for making this joint research possible. The authors would also like to thank the Computational Learning and Motor Control (CLMC) lab *Learning locomotion team* from the University of Southern California (USC) for their valuable input. Also Michael Bloech from the Autonomous System Lab (ASL) of ETH Zurich for his help in state estimation. The authors would like to thank the colleagues that collaborated for the success of this project: Bilhal Rehman, Hamza Khan, Jake Goldsmith, Marco Frigerio, Victor Barasuol and our team of technicians.

References

[1] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of HyQ – a hydraulically and electrically actuated quadruped robot," *Journal of Systems and Control Engineering*, 2011,

[2] E. Muybridge, *Animals in motion*. Courier Dover Publications, 1957.

[3] R. McGhee and A. Frank, "On the stability properties of quadruped creeping gaits," *Mathematical Biosciences*, vol. 3, no. 0, pp. 331 – 351, 1968,

[4] J. Estremera and P. G. De Santos, "Free gaits for quadruped robots over irregular terrain," *The International Journal of Robotics Research*, vol. 21, no. 2, pp. 115–130, 2002,

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

[6] V. Barasuol, J. Buchli, C. Semini, M. Frigerio, E. De Pieri, and D. Caldwell, "A reactive controller framework for quadrupedal locomotion on challenging terrain," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2013,

[7] I. Havoutis, C. Semini, J. Buchli, and D. G. Caldwell, "Quadrupedal trotting with active compliance," *IEEE International Conference on Mechatronics (ICM)*, 2013,

[8] M. Focchi, V. Barasul, I. Havoutis, J. Buchli, C. Semini, and D. G. Caldwell, "Local reflex generation for obstacle negotiation in quadrupedal locomotion," *International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)*, 2013,

[9] J. R. Rebula, P. D. Neuhaus, B. V. Bonnlander, M. J. Johnson, and J. E. Pratt, "A controller for the littledog quadruped walking on rough terrain," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1467–1473,

[10] D. Pongas, M. Mistry, and S. Schaal, "A robust quadruped walking gait for traversing rough terrain," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1474–1479,

[11] J. Z. Kolter, M. P. Rodgers, and A. Y. Ng, "A control architecture for quadruped locomotion over rough terrain," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 811–818,

[12] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," in *International Journal of Robotics Research (IJRR)*, no. 2, 2010, pp. 236–258,

[13] M. Zucker, N. Ratliff, M. Stolle, J. Chestnutt, J. A. Bagnell, C. G. Atkeson, and J. Kuffner, "Optimization and learning for rough terrain legged locomotion," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 2, pp. 175–191, Feb. 2011,

[14] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, "Bounding on rough terrain with the littledog robot," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 2, pp. 192–215, Feb. 2011,

[15] P. Vernaza, M. Likhachev, S. Bhattacharya, A. Kushleyev, and D. D. Lee, "Search-based planning for a legged robot over rough terrain," in *IEEE International Conference on Robotics and Automation*, 2009,

[16] I. Havoutis, J. Ortiz, S. Bazeille, V. Barasuol, C. Semini, and D. G. Caldwell, "Onboard perception-based trotting and crawling with the hydraulic quadruped robot (HyQ)," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013,

[17] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011, pp. 127–136,

[18] S. Schaal, "The sl simulation and real-time control software package," Tech. Rep., 2009,

[19] M. Kalakrishnan, J. Buchli, P. Pastor, and S. Schaal, "Learning locomotion over rough terrain using terrain templates," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009, pp. 167–172,

[20] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959,

[21] J. Pratt, C.-M. Chew, A. Torres, P. Dilworth, and G. Pratt, "Virtual model control: An intuitive approach for bipedal locomotion," *International Journal of Robotics Research (IJRR)*, 2001,

[22] M. Bloesch, M. Hutter, M. Hoepflinger, S. Leutenegger, C. Gehring, C. Remy, and R. Siegwart, "State estimation for legged robots - consistent fusion of leg kinematics and imu," in *Robotics: Science and Systems Conference (RSS)*, 2012,