Journal of Intelligent & Robotic Systems (2019) 95:19–46
https://doi.org/10.1007/s10846-018-0835-3

Optimization of Vehicle Mounting Motions and Its Application to Full-Sized Humanoid, DRC-Hubo

Kiwon Sohn[1]

Received: 19 September 2017 / Accepted: 4 April 2018 / Published online: 27 April 2018

Abstract

This paper describes optimization of humanoid's whole body motion for vehicle mounting task. To accomplish the goal, a trajectory optimization framework based on the reinforcement learning agent is used in this study. The guideline input trajectory is planned and optimized as regards various dynamic and kinematic limitations of humanoid in the framework. In previous studies, the authors demonstrated test-and-evaluation process of the framework using a full-sized humanoid, Hubo+. Experimental testing however presented several problems like overheating and self-collisions. To resolve those issues and to validate the trajectory optimization approach, another humanoid, called DRC-Hubo, is newly designed. Keeping a main structure of the optimization framework, the cost value functions are revised to meet dynamic and kinematic changes. Experimental test and verification process using a simulation model and a physical prototype is demonstrated to confirm the efficacy of the trajectory optimization approach. For both processes, two different types of ground vehicle are used. Last, analytical comparisons with other techniques are also conducted for validation of the proposed framework.

Keywords Humanoids · Vehicle mounting · Optimization · Reinforcement learning · DRC-Hubo · Darpa robotics challenge

1 Introduction

In disaster management, letting workers to perform emergency operations within the early hours is critical for mitigation. However, some areas are highly dangerous for relief team to get inside [1]. The Fukushima Daiichi nuclear meltdown is the case that explosions would have been prevented if more capable disaster response robots got into the reactor and ventilated the gas. Today's humanoids however rarely have enough power capacity to move such long distances. Furthermore, they often do not have robust locomotion skills to pass obstacles in the site. As such, enabling robots to drive a vehicle autonomously and to move quickly was identified by DARPA as an important task [2]. A robot that drives vehicles can quickly reach the scene of the accident and perform emergency operations.

Currently, it is quite uncommon for humanoids to posses the enough torque, range-of-motion and balancing capability to get inside (ingress) a vehicle without human's help. Not only that, present humanoids rarely have the perception and cognition to do autonomous driving. Therefore, there is intellectual merit to empower a humanoid to handle vehicles. Vehicle handling requires humanoids to have the advanced perception, cognition, control and motion planning algorithms. This task also requires optimization of trajectories (humanoid's motion) concerning different time-variant properties. The DARPA Robotics Challenge, announced in 2012 [3], further emphasizes the importance of vehicle handling. Vehicle handling was assigned as the first mission among the eight required tasks for all participants in the competition.

There are also broad impacts and commercial merits if humanoids can drive conventional vehicles. First, as a general purpose machine, humanoids can be programmed to drive vehicles and also to carry out various material handling tasks (such as valve turning or hose attachment). To remove toxic materials or to rescue human beings in disaster, such capability (carrying out various tasks) is highly important. However, driverless cars cannot execute such material handling tasks. Second, the braking and driving mechanisms should be physically altered and various actuators and

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10846-018-0835-3) contains supplementary material, which is available to authorized users.

✉ Kiwon Sohn
skw1125@gmail.com

[1] Electrical and Computer Engineering Department, University of Hartford, West Hartford, CT 06117, USA

sensors should be added to make a vehicle unmanned. This entails cost and raises engineering effort (since individual factory lines are necessary to make different kinds of unmanned vehicles). However, if humanoid robots can drive existing vehicles, there is no need to modify them. The robots can be re-programmed to drive a wide variety of utility vehicles (such as forklifts, trucks, jeeps and tractors).

This paper particularly describes planning of humanoid's whole body motion to mount and unmount (ingress and egress respectively) ground vehicles. They are the initial and last steps of vehicle handling task. They demand whole body balance, collision handling and energy consumption minimization. The DRC-Trials, held in 2013 [4] and the DRC-Finals, held in 2015 [5] demonstrate difficulties of vehicle mounting as well as driving. In the challenges, only a few robots passed the finish line in the driving course. Many teams even could not control their robot to start the vehicle [6]. In Trials, a team who completed the vehicle driving mission within just 5 minutes also did not attempt mounting [7].

For vehicle handling, the supervised control has been often used as a direct and simple solution. In [8], HRP-1 drove the forklift and the backhoe-cockpit respectively through tele-operation. The limbs of HRP-1 were controlled through human operator's command. Vehicles however have many features like stick shifts, control pedals, and handwheel. The robot should maneuver around them while mounting. Nonetheless, the papers did not address the issues and did not describe the approaches for the problem. Furthermore, wireless communication is often unstable in outdoors and the signal can drop out entirely. Though wired-tethering can be used for resolving these issues, the robot's mobility can be impeded by long cables.

As such, humanoids should be able to execute tasks by themselves. However, the tasks are often increasingly complex. Therefore, robots need motion-planning which can be used for their high degree-of-freedom (DOF) kinematic structure. The most common method to deal with high dimensional search spaces is to use probabilistic sampling [9], such as Rapidly-exploring Random Trees. RRT algorithms can produce a collision-free (both internal and external) trajectory in the humanoid's configuration space while guaranteeing quasi-static balancing. However, they do not account for important factors such as dynamic characteristics and energy efficiency. Such sampling methods also have many difficulties in checking the robot's balance when it switches from the one foot to another. An additional technique for complex motion planning is to utilize recorded motion data from human subjects. It is more direct approach and the recorded data generates more human-like movements [10]. However, the motion-capture based approaches often do not take account of important factors such as balancing, collision-avoidance and kinematic differences.

This paper adopts an alternative approach to plan and optimize vehicle mounting motions. The authors used a reinforcement learning agent based optimization method. The agent selects a states sequence which is optimal as regards given dynamic and kinematic constraints. In previous study [11], the authors introduced the framework which was especially designed for humanoid's upper-body motion for material handling. The authors followed up this work by adding multiple time-varying constraints which are necessary for humanoid vehicle mounting motions [12, 13]. In the studies, the reinforcement learning agent converted input raw paths to output trajectories which have the minimum penalty values. Then, the optimized trajectory was examined in simulation environment and physical experiments with Hubo+[1]. Experimental testing using Hubo+ however issued several problems such as overheating and self-collision in the studies. These issues were mainly due to Hubo+'s short limbs and limited torques of multiple joints.

To resolve the issues, technical requirements were specified for the new Hubo+ design. They include changes of the joint torques and the limb lengths. Figure 1 demonstrates a humanoid which is upgraded based on the new technical specifications. It is called DRC-Hubo. This paper focuses on planning and optimizing the newly built humanoid's motions, especially for vehicle mounting task. Keeping overall structure of the optimization framework in previous studies [11, 13], cost value functions are updated to account for dynamic and kinematic changes between Hubo+ and DRC-Hubo. For the experimental evaluation of the presented framework, the optimized trajectory is tested to the virtual model in a simulation environment, OpenRAVE (OpenHubo) [14]. Then, through the real physical platform, the trajectory optimization framework is verified with the effectiveness for humanoid's vehicle mounting motion design.

As such, this paper presents the first successful application of the reinforcement learning agent based trajectory optimization technique for the physical platform. Both test-evaluation and verification process which demonstrate vehicle-mounting motions of the full-sized humanoid are newly provided using two different conventional vehicles. Analytical comparisons with other techniques are also implemented to underscore the validation. This paper also introduces the optimization of vehicle-robot interface motions.

First, Section 2 describes electrical and mechanical issues of Hubo+ and shows critical design updates of DRC-Hubo based on the addressed technical requirements. Section 3 presents a building procedure of the trajectory

[1]Hubo+ (released in 2011) is the generation following the 2007 KHR-4 Hubo

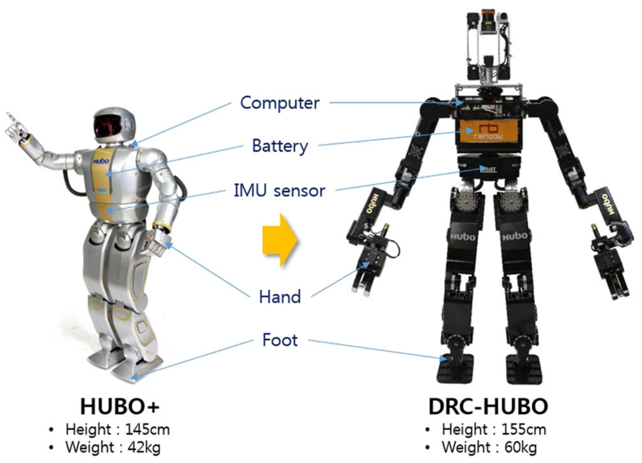


Fig. 1 Hubo+ and DRC-Hubo

optimization framework for the new model. Section 3.1 shows an overview of the reinforcement learning agent and Section 3.2 presents modified cost value functions which reflect new dynamic and kinematic constraints of DRC-Hubo. Sections 4 and 5 then demonstrate experimental results with a golf cart and a utility ground vehicle respectively. Both sections describe test-and-evaluation process using OpenHubo and verification process using DRC-Hubo. Section 6.1 shows analytical data from comparison with other techniques which were tested to confirm the efficacy of the proposed framework. Section 6.2 demonstrates extended studies of the presented framework. Last, Section 7 concludes with future works.

2 Technical Design Changes of Hubo

As addressed in Section 1, short legs and limited joint torques of Hubo+ issued several problems such as self-collision and overheating during vehicle mounting motions.

Because of its heavy weights of each leg, the rigidity of each link pose become worsen with a wrench effect on the movement of Hubo+. It resulted in unanticipated internal collisions which occurred between the connected bodies of several joints, specifically hip and ankle. By adding kinematic constraints in the joint angle, the connected link can be avoided from colliding with other parts. However, the short leg length of Hubo+ did not allow the further decrease of the joint angle limitation especially when climbing the high vehicle floor.

Some joints which have high torques also created overheating in the Hubo+'s power distribution system. The applied torques satisfied the hardware limits of the devices such as harmonic-drive gears and brushless dc motors. However, heats were accumulated by the high current in the main circuit board and eventually generated burnouts. Especially, due to its short leg length, it occurred through the latest half phase of ingress when Hubo+ stayed on knee bent pose for long period.

To resolve the issues, several technical requirements are specified for the next robot design. Figure 2 describes the newly addressed requirements such as increment of leg lengths, limits in joint angle and torque.

Figure 3 shows the new design features of DRC-Hubo which are generated based on the addressed technical requirements above. In the updated model, the robot's joint angle and torque limits become increased and its legs and arms are extended. Each arm has a more powerful gripper hand (changeable end-effector) and 7-DOF design. In each ankle and wrist joint, a 3-axis FT (force-torque) sensor are attached with increased sensing ranges. Battery capacity also got increased for more durable task implementation capabilities. These kinematic and dynamic updates of DRC-Hubo are presented in Section 3.2 with more details.

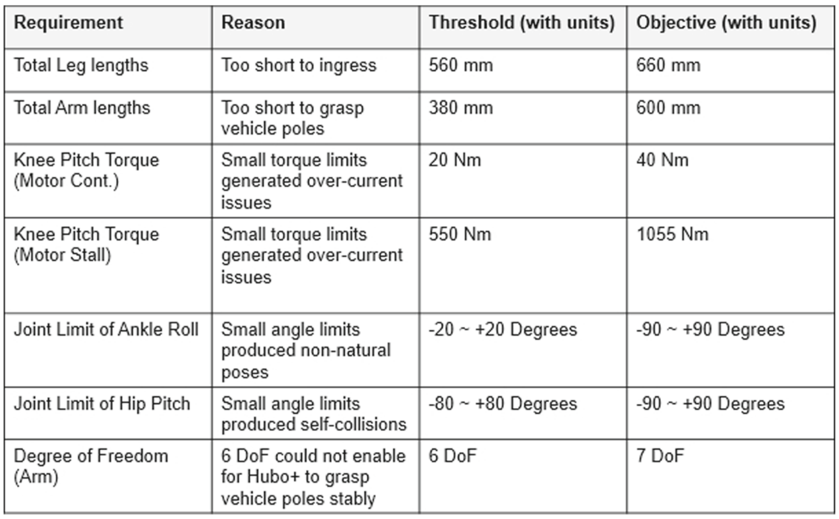
3 Trajectory Optimization Framework

It is not trivial to enable a full-sized humanoid to climb up (mount) on a utility vehicle. There are several unknown factors, such as air pressure (tire) or suspension level, that makes it difficult to model the vehicle accurately. Not only that, there may be some difference in shock absorber and tire even for the same vehicle type. As such, it is not simple to design ingress (or egress) motion by planning dynamic movement.

The dynamic parameters can be bounded to make it possible to generate dynamic motions when the vehicle type is determined. However, in this study, to present more general approach which even can be adaptively applied for different types of vehicles, motion planning is implemented in quasi-static mode to be stable enough to compensate factors which are not known.

Such planned motions have the robot's discrete configurations which meet all given static constraints on pre-arranged time frames. They are called key frames in this study and are initially designed from various path-planners considering its importance in the applied motion (such as foot-landing or lifting) on discrete time step. The key frames which are not enough dense can obey time-varying requirements such as joint velocity or acceleration limitations. In this study, 100 ms time step was chosen as a sampling frequency to get the discrete frames from the initially designed trajectory. It is the maximum value which satisfies the requirement above [13].

However, the initial frames from path-planners often do not satisfy constraints which are important for vehicle mounting task as described in Section 1. In previous studies [11, 13], the whole body motion optimization framework



Requirement	Reason	Threshold (with units)	Objective (with units)
Total Leg lengths	Too short to ingress	560 mm	660 mm
Total Arm lengths	Too short to grasp vehicle poles	380 mm	600 mm
Knee Pitch Torque (Motor Cont.)	Small torque limits generated over-current issues	20 Nm	40 Nm
Knee Pitch Torque (Motor Stall)	Small torque limits generated over-current issues	550 Nm	1055 Nm
Joint Limit of Ankle Roll	Small angle limits produced non-natural poses	-20 ~ +20 Degrees	-90 ~ +90 Degrees
Joint Limit of Hip Pitch	Small angle limits produced self-collisions	-80 ~ +80 Degrees	-90 ~ +90 Degrees
Degree of Freedom (Arm)	6 DoF could not enable for Hubo+ to grasp vehicle poles stably	6 DoF	7 DoF

Fig. 2 Technical design requirements

was presented for optimizing the frames as regards time-variant constraints of vehicle mounting task (ingress and egress). The optimization followed next four steps: 1) a selected path-planner initially designs guideline input trajectories for hip and end-effectors (in task space); 2) key frames are extracted from the input and they formed the multiple sets of candidate trajectory with their neighbors; 3) a grading of the each frame is executed considering all the given kinematic and dynamic constraints; 4) the most optimal frames are searched and converted to the joint trajectories (in configuration space) as a final outcome.

As such, in the framework, the guideline input trajectory consists of finite states (key frames) which each needs to be optimized with multiple constraints. Therefore, the reinforcement learning agent is used to interact with cost function modules (which are defined from all the given constraints) and to grade each key frame efficiently. Section 3.1 shows an overview of the reinforcement learning agent.

In this study, the overall structure of the trajectory optimization framework is kept. However, its cost function modules become modified to reflect newly added dynamic and kinematic constraints of DRC-Hubo. Section 3.2 shows the process and presents more details of kinematic and dynamic changes in Hubo model.

3.1 Reinforcement Learning Agent

Figure 4 demonstrates the whole body motion trajectory optimization framework. It is built based on reinforcement learning and designed for the vehicle mounting task. A

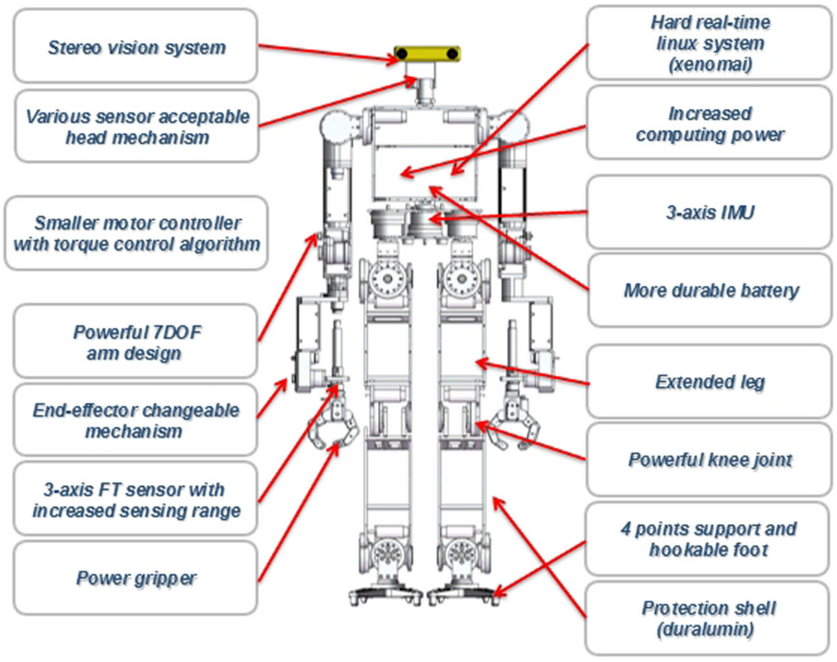

Stereo vision system
Various sensor acceptable head mechanism
Smaller motor controller with torque control algorithm
Powerful 7DOF arm design
End-effector changeable mechanism
3-axis FT sensor with increased sensing range
Power gripper
Hard real-time linux system (xenomai)
Increased computing power
3-axis IMU
More durable battery
Extended leg
Powerful knee joint
4 points support and hookable foot
Protection shell (duralumin)

Fig. 3 Critical design changes of DRC-Hubo [15]

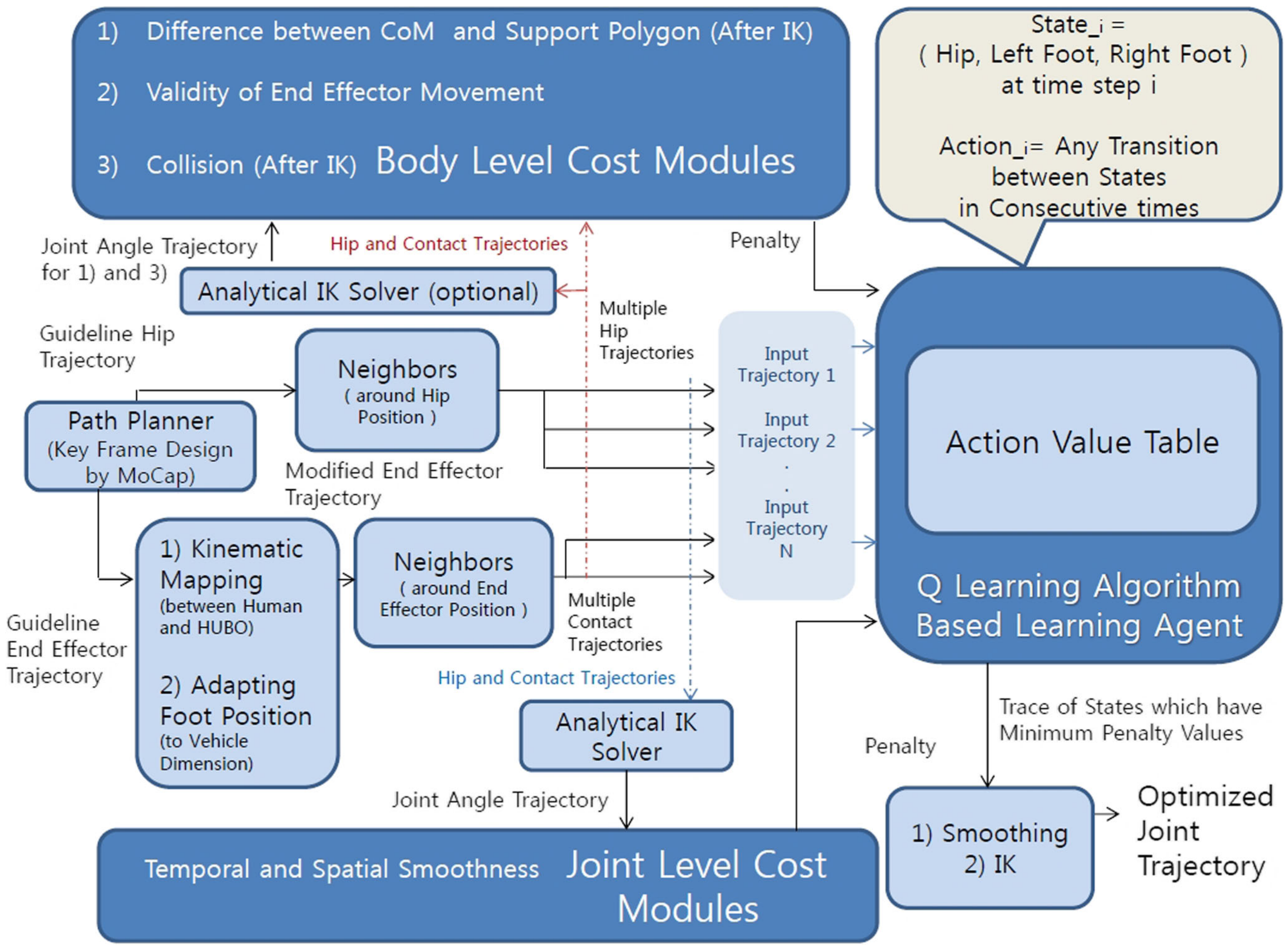
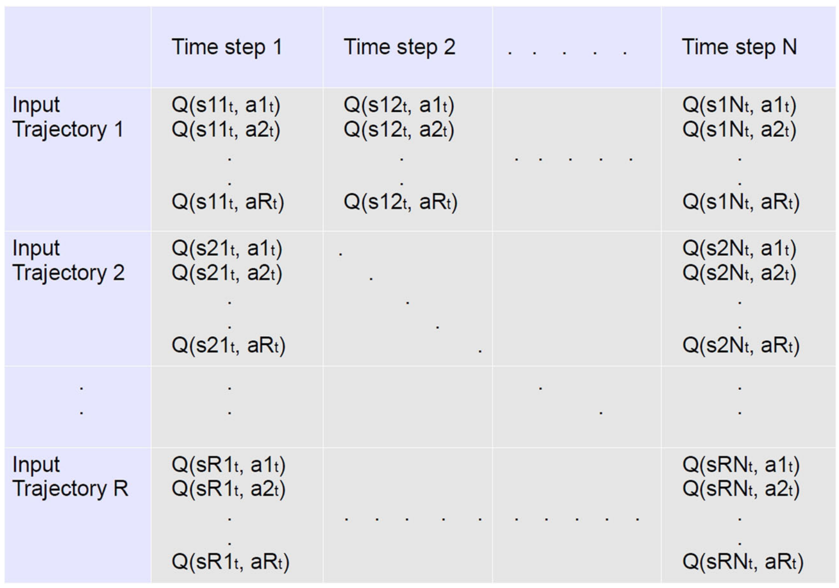

1) Difference between CoM and Support Polygon (After IK)
2) Validity of End Effector Movement
3) Collision (After IK)
Body Level Cost Modules
State_i = (Hip, Left Foot, Right Foot) at time step i
Action_i= Any Transition between States in Consecutive times
Joint Angle Trajectory for 1) and 3)
Hip and Contact Trajectories
Penalty
Analytical IK Solver (optional)
Multiple Hip Trajectories
Guideline Hip Trajectory
Neighbors (around Hip Position)
Path Planner (Key Frame Design by MoCap)
Modified End Effector Trajectory
Input Trajectory 1
Input Trajectory 2
Input Trajectory N
Action Value Table
Q Learning Algorithm Based Learning Agent
1) Kinematic Mapping (between Human and HUBO)
2) Adapting Foot Position (to Vehicle Dimension)
Guideline End Effector Trajectory
Neighbors (around End Effector Position)
Multiple Contact Trajectories
Hip and Contact Trajectories
Analytical IK Solver
Trace of States which have Minimum Penalty Values
Penalty
Joint Angle Trajectory
1) Smoothing 2) IK
Optimized Joint Trajectory
Temporal and Spatial Smoothness
Joint Level Cost Modules

Fig. 4 *Q* learning agent based trajectory optimization framework [13]

Fig. 5 *Q* table at learning time t [11]



	Time step 1	Time step 2	. . . . .	Time step N
Input Trajectory 1	$Q(s11_t, a1_t)$ $Q(s11_t, a2_t)$ . . $Q(s11_t, aR_t)$	$Q(s12_t, a1_t)$ $Q(s12_t, a2_t)$ . . $Q(s12_t, aR_t)$	. . . . .	$Q(s1N_t, a1_t)$ $Q(s1N_t, a2_t)$ . . $Q(s1N_t, aR_t)$
Input Trajectory 2	$Q(s21_t, a1_t)$ $Q(s21_t, a2_t)$ . . $Q(s21_t, aR_t)$	. . . . .		$Q(s2N_t, a1_t)$ $Q(s2N_t, a2_t)$ . . $Q(s2N_t, aR_t)$
. .	. .		. .	. .
Input Trajectory R	$Q(sR1_t, a1_t)$ $Q(sR1_t, a2_t)$ . . $Q(sR1_t, aR_t)$	. . . . .	. . . . .	$Q(sRN_t, a1_t)$ $Q(sRN_t, a2_t)$ . . $Q(sRN_t, aR_t)$

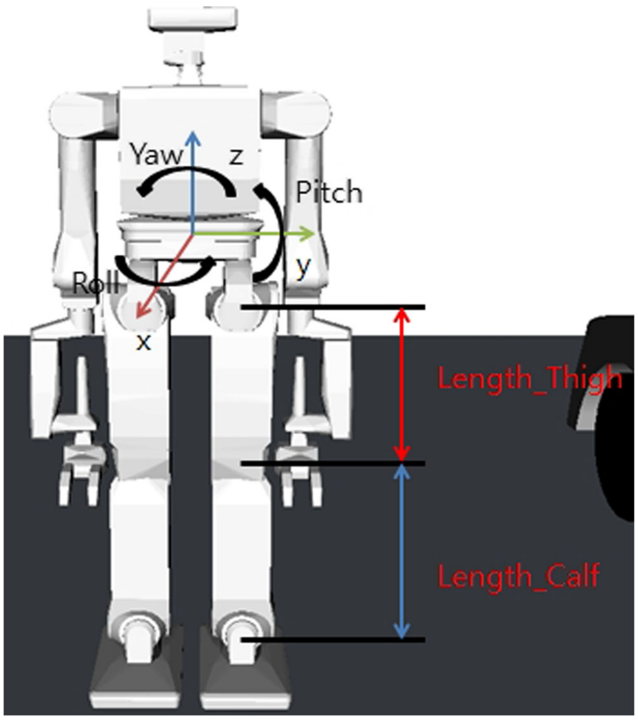


Fig. 6 DRC-Hubo's body fixed coordinate

path-planner module was used to initially design the sub-optimal hip and end-effector trajectories in this framework. Vehicle mounting demands whole body balance, and handling both external and self-collisions. The physically impaired or the elderly demonstrates similar difficulties with humanoids which also have limbs of limited range-of-motion, force and torque. As such, observation of people who constrain limb positions during ingress can deliver applicable source trajectory to humanoids. Therefore, in the presented framework, the path-planner module first generated initial trajectories (especially hip and end-effector position) from the recorded motion of a human subject in motion-capture system (mocap).

Based on the pre-chosen frequency, the key frames were then extracted from the trajectories. The frequency value was selected as the maximum value (to reduce amount of key frames) while not violating time-variant constraints (10 Hz is used in this study). Sohn and Oh [13] presents more details of the sampling frequency determination in the path-planner module. The net result is that guideline key frame trajectories are generated as an outcome of the path-planner module in the framework.

However, the primary goal of the path-planning module is just to generate initial contact and pelvis position sequences for the specified task. Therefore, other search-based sampling methods (for example, RRT and its variants [16]) can be also applied to the optimization framework considering conditions and features of the task. As described in Section 1, even such state of the art path-planning methods which are tailored to humanoid platform often do not account for important factors such as balancing during foot-switch or spatial smoothness. As such, they need optimization process [17, 18] to satisfy the required constraints of the given tasks. Sohn and Oh [11] demonstrates that guideline trajectories from RRT also can be successfully processed in the presented framework.

When it is necessary, a post-processing was implemented on the guideline trajectories for kinematic re-mapping between humanoid and the captured bodies. During the

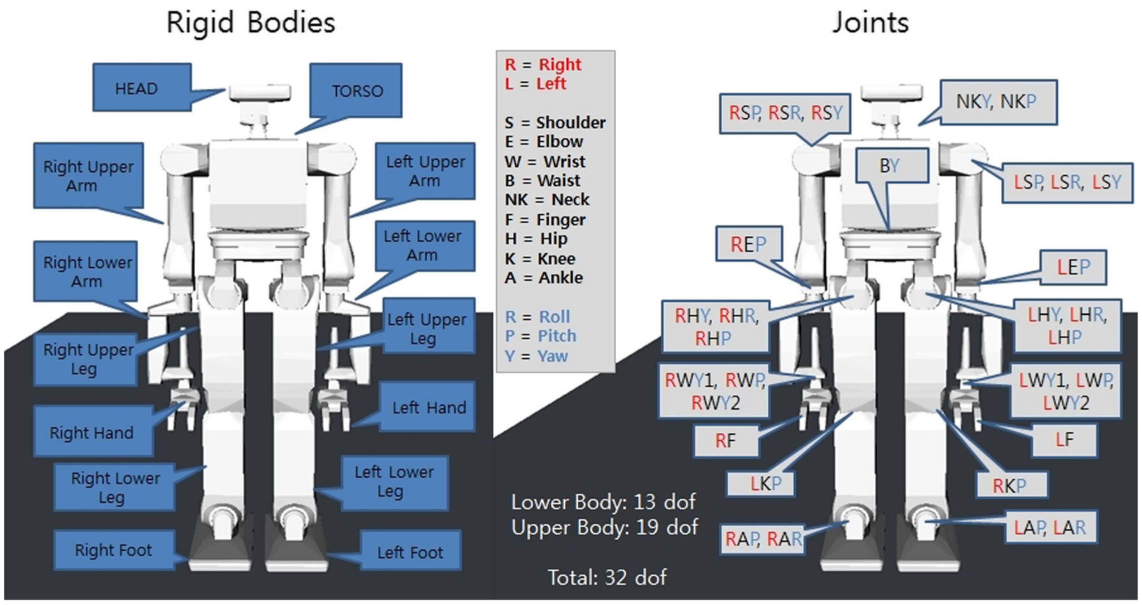


Fig. 7 Links and joints of DRC-Hubo

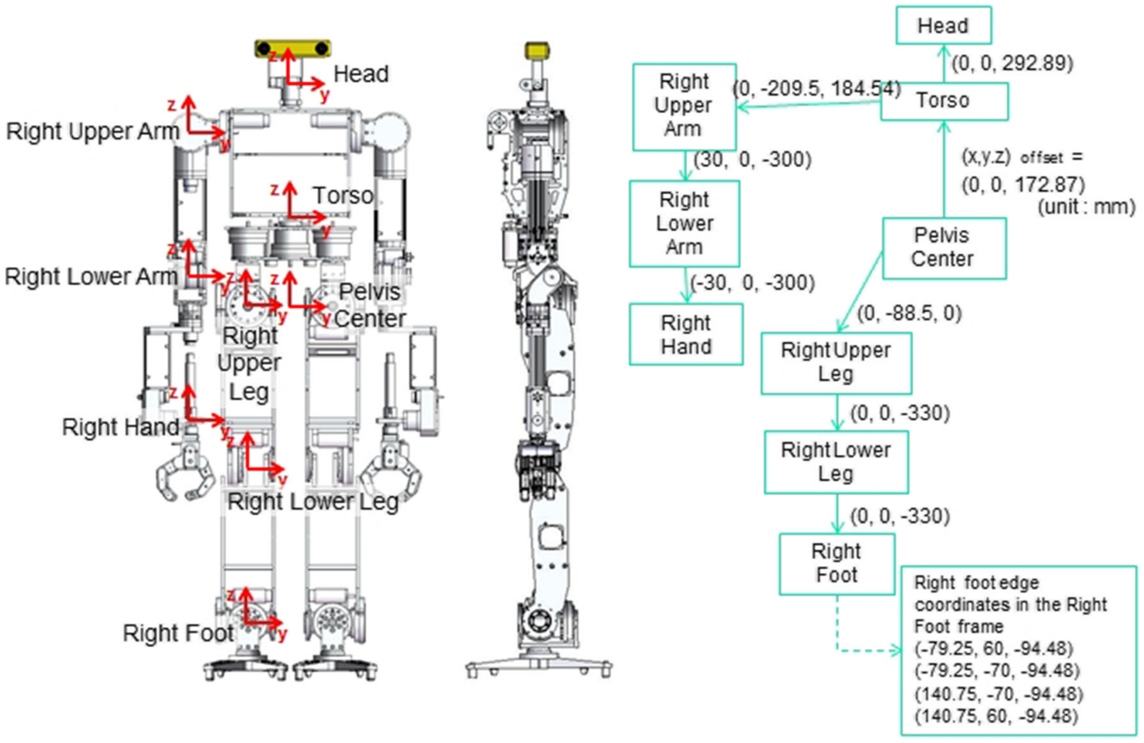


Fig. 8 Local coordinate for each link of DRC-Hubo [15]

stage, the vehicle's kinematic dimension also redesigned the initial trajectories to make the capture motion fit the vehicle's kinematic attributes more suitably. More details with the initial path-planning and post-processing stages are described in [12, 13].

After the post-processing stage, within limited boundaries, the guideline hip and end-effector trajectories look through their neighbors for each time step. At each step, sets of neighboring points were formed by merging all the searched point. The built sets became states in the state-action value table (called Q table) for the corresponding time step. As such, sets in each time step became states in the same column of the Q table. This process was repeated till the Q table's end column which means the last time step of the guideline key frame trajectories. In such a way, the hip and end-effector trajectories were transfered to the learning agent as input states with neighboring points.

Figure 5 illustrates the Q table at learning time t. When $P1$ hip points and $P2$ individual end-effector (each feet in this case) points were extracted from the corresponding neighbors of the guideline key frame trajectories, there are $P1 \times P2(\text{left}) \times P2(\text{right}) \times N$ different states (which is R in the Q table of Fig. 5). For sampling human motion-capture, 10 Hz is used in this study as described above. This indicates that $N\times$ 100 ms will be needed to finish the planned trajectory. As such, the time length of initially designed motions is highly related to the total dimension of state-and-action domain. The neighboring bounds selection process which defines the exploration potential (of the presented approach) also directly affects the dimension of the domain.

The state-building process above and its effects on computational complexity are further discussed in [13]. Through the analysis of interaction between exploration power and computation cost of the approach, the optimized values for sampling time and boundary limits of neighbor search process were determined in the study.

In the Q table of Fig. 5, state sij_t is a set of hip and individual end-effector (foot in this study) positions which is extracted from the i^{th} neighbor set on the time step j. The table also presents that each state has the particular set of actions, a_t. When sij_t is the current state, the allowed action for the state is a movement between sij_t and any state in the subsequent time step $j+1$. For example, if the current state is sij_t and an_t is a selected action for the state, the next state becomes $snj+1_t$. The net result is that there are finite number of states over discrete time steps and the decision making processor can choose one of any actions that are available in each state.

It means that the Q table above meets all the demands of Markov Decision Process. Therefore, the generated table can be graded by the temporal difference method, Q learning algorithm [19]. The algorithm finds an optimal

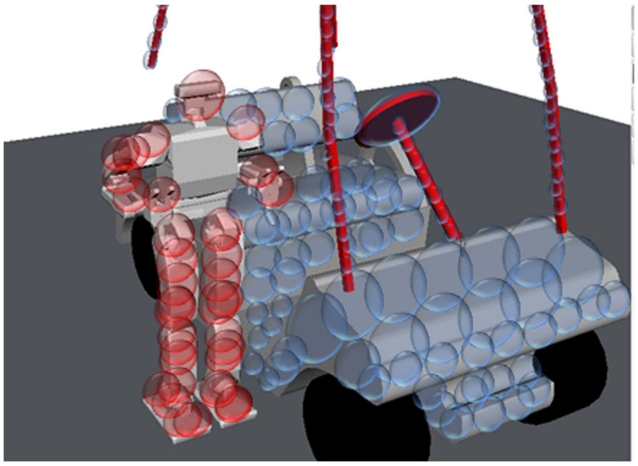


Fig. 9 Bounding spheres in DRC-Hubo and golf cart for collision checking

action-selection policy for the given Q table. Equation 1 shows how Q values in the table of Fig. 5 are updated.

$$Q(sij_t, an_t) \leftarrow Q(sij_t, an_t) + \alpha_t(sij_t, an_t)* \tag{1}$$

$$[P_{t+1} + \gamma \min_{a_{t+1}} Q(snj + 1_{t+1}, a_{t+1}) - Q(sij_t, an_t)]$$

α is the learning rate and P is the penalty (cost) value. γ is a discount factor for the minimum future Q value.

The cost values of each state-and-action pair in the Q table were computed based on its dynamic and kinematic costs. Since the initial trajectory is designed from the path-planner module in this study, the bins in the end column of the table have the previously known states. It means that it is possible to grade the Q table reversely from the end column to the first one. This approach efficiently reduced the grading time. There are no future states for each bin in the end column. Therefore, their Q values are calculated solely from the present cost values which are fixed regardless of number of the updates. It resulted in constant minimum future Q value for every bin in the former time step. As such, the Q values are not diverged by future cost factors. The net result is that only one iteration is necessary for the Q table grading to make the optimal trajectory. More details with the inverse-grading of Q table are given in [12, 13].

3.2 Cost Function Modules and Output Trajectory Generation

To grade each bin (state-and-action pair) of the Q value table with learning algorithm, multiple constraints such as physical limits of humanoids are considered as penalties in the optimization framework. For this, various cost (penalty) functions are determined to meet updated dynamic and kinematic requirements of DRC-Hubo. The cost values were generally classified into two groups which are called the robot body level and the joint level. The robot body level cost functions checked the static balance of the robot, cogency of end-effector motion and collisions. The joint level cost functions measured the configuration space penalties such as joint angle, velocity and torque limit obedience. For this, joint angles (configuration space) were computed first based on the selected bins in the Q table through IK process.

Figures 6 and 7 demonstrate DRC-Hubo's body fixed coordinate with its all joints (32-DOF) and links. While Hubo+ model has 5-DOF for each hand, DRC-Hubo's hand has only 1-DOF. Three fingers of each hand are actuated by only one DC motor in DRC-Hubo model.

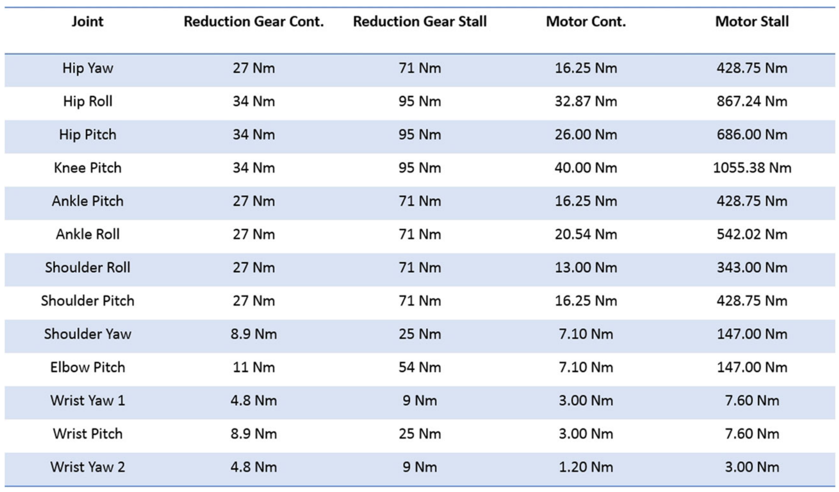
First, the cost values for newly upgraded kinematic features of DRC-Hubo are determined at the robot's body level. The static balancing capability of the robot, collisions and validity of end-effectors motion are measured in the level.

Each bin in the Q table was initially tested whether a static balancing posture can be derived. To check the feature, the center-of-mass (CoM) position of DRC-Hubo was calculated. Each hip and end-effector point (of the state in Q table) has 3-DOF position and 1-DOF orientation values. From the values, the analytical IK process calculated the joint angles. Figure 8 demonstrates the local coordinate for each link of DRC-Hubo which is used for the process.

The DRC-Hubo's CoM position was then calculated from the joint angles through the forward-kinematics (FK) process (considering mass of each link and pre-determined foot position which touches ground) and checked whether it resides inside support polygon. The comparison determined the static balance of the pose. The bin in Q table which does not satisfy the criterion received a penalty value during grading.

For computation of collisions, the bounding volume hierarchy is applied. Figure 9 demonstrates bounded spheres which are serialized in the ground vehicle and the DRC-Hubo model. To detect collisions, L^2 distances between two spheres (each extracted from vehicle and DRC-Hubo) were calculated for every possible pair. The pair was considered to be collision-free when the computed distance is longer than the sum of each radius. A similar approach was also implemented on two sampled spheres of the robot for self-collision checks. Pairs in Q value table which are not collision-free received a penalty cost either.

Each bin in the Q table was also tested whether it has valid movements of end-effectors. For examples, the bin which has displacements of supporting foot was assigned a penalty. Any action which has attempted movements of the foot can yield unnecessary contact friction and such movement can cause the supporting foot to break contact



Joint	Reduction Gear Cont.	Reduction Gear Stall	Motor Cont.	Motor Stall
Hip Yaw	27 Nm	71 Nm	16.25 Nm	428.75 Nm
Hip Roll	34 Nm	95 Nm	32.87 Nm	867.24 Nm
Hip Pitch	34 Nm	95 Nm	26.00 Nm	686.00 Nm
Knee Pitch	34 Nm	95 Nm	40.00 Nm	1055.38 Nm
Ankle Pitch	27 Nm	71 Nm	16.25 Nm	428.75 Nm
Ankle Roll	27 Nm	71 Nm	20.54 Nm	542.02 Nm
Shoulder Roll	27 Nm	71 Nm	13.00 Nm	343.00 Nm
Shoulder Pitch	27 Nm	71 Nm	16.25 Nm	428.75 Nm
Shoulder Yaw	8.9 Nm	25 Nm	7.10 Nm	147.00 Nm
Elbow Pitch	11 Nm	54 Nm	7.10 Nm	147.00 Nm
Wrist Yaw 1	4.8 Nm	9 Nm	3.00 Nm	7.60 Nm
Wrist Pitch	8.9 Nm	25 Nm	3.00 Nm	7.60 Nm
Wrist Yaw 2	4.8 Nm	9 Nm	1.20 Nm	3.00 Nm

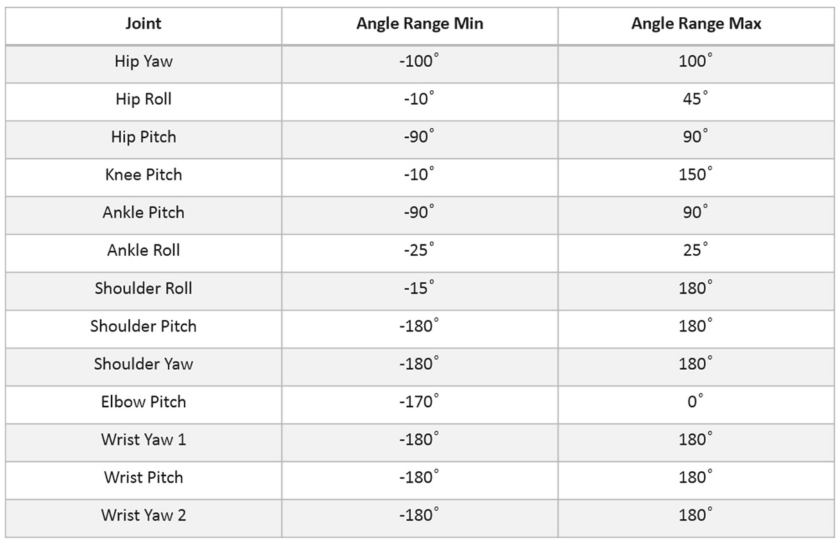
Fig. 10 DRC-Hubo's joint torque [15]

prematurely. More details with the cost functions in the body level is presented in [13].

The cost functions in the joint level accounted for dynamic features of DRC-Hubo. First, torque values of every joint were calculated to predict energy consumption for each bin in the Q table. ProPac [20], a Mathematica package, was applied to relate the joint angles and torques. It computed all components of the Poincare' equation. Under a given trajectory, torque value for each joint was calculated using the forward-kinematic model (built based on the collected joints and links data and DRC-Hubo's joint hierarchy) and the inverse-dynamics method. When there exists a bin which has an action that does not obey the defined hardware limits (the maximum torque of the joint), it received a penalty. It made the generated motion does not require too much torque or high current.

Bins (in the Q table) which exceeds temporal and spatial smoothness limits (of joint angle, velocity and torque) also got a penalty. The penalties guaranteed that the optimized motion from the presented framework satisfies the dynamic and kinematic limits of the robot's each joint. Limiting joint angle and its velocity ensured a smooth trajectory over time (temporal smoothness) and penalizing actions which exceed the maximum rate of torque change (calculated from inverse dynamics method above) ensured spatial smoothness. Figures 10 and 11 illustrate DRC-Hubo's dynamic and kinematic limits of each joint. More details with the cost functions in the joint level is presented in [11, 13].

As such, if a bin of the Q table has a pair of state-and-action $((sij_t, an_t))$ which does not meet certain sets of constraint (in body and joint level), penalty values was assigned to the bin from the corresponding cost function modules. In reinforcement learning, different weights on penalties can result in different output trajectories. Therefore, the weighting factor value on each cost function module was chosen considering the relative importance of the corresponding penalty.



Joint	Angle Range Min	Angle Range Max
Hip Yaw	-100°	100°
Hip Roll	-10°	45°
Hip Pitch	-90°	90°
Knee Pitch	-10°	150°
Ankle Pitch	-90°	90°
Ankle Roll	-25°	25°
Shoulder Roll	-15°	180°
Shoulder Pitch	-180°	180°
Shoulder Yaw	-180°	180°
Elbow Pitch	-170°	0°
Wrist Yaw 1	-180°	180°
Wrist Pitch	-180°	180°
Wrist Yaw 2	-180°	180°

Fig. 11 DRC-Hubo's Joint Angle Range [15]

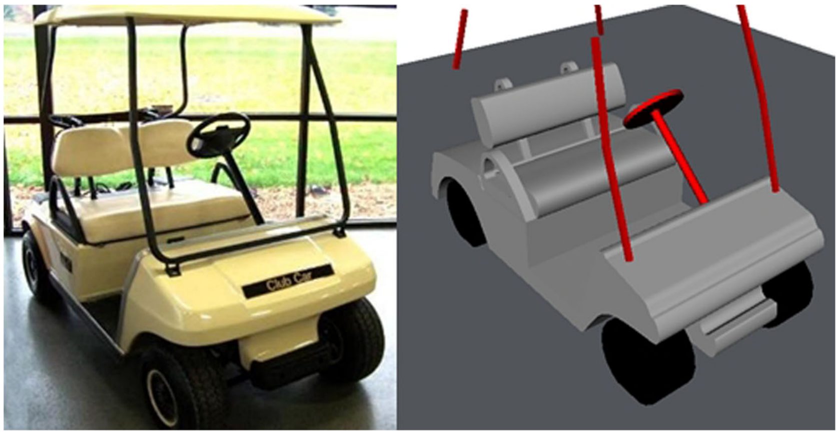


Fig. 12 Vehicle 1: Club Car DS IQ (2008) and KinFu model

3.3 Output Trajectory Generation

The Q value table was updated according to Q value iteration equation [19] with penalty values P from various cost functions in both joint and body levels. After grading, the bin which has a state-and-action pair with the minimum cost value was explored at each time step. The states of the searched pairs became new key frames of the applied

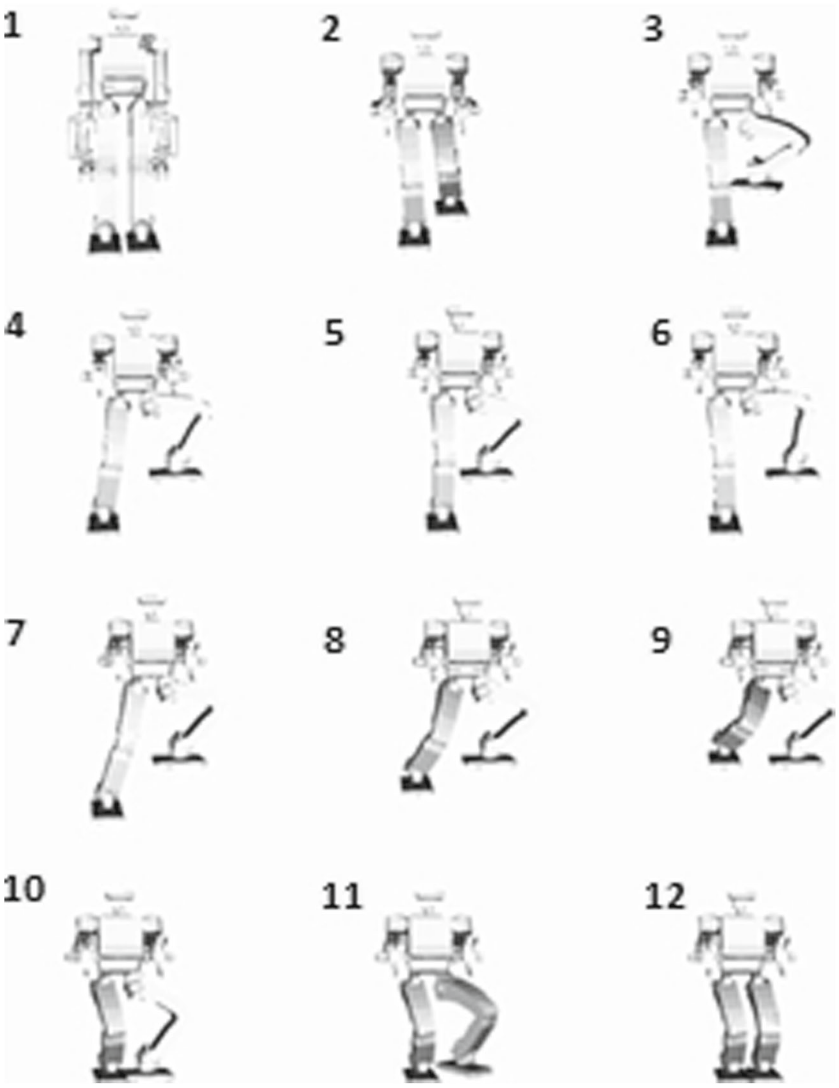


Fig. 13 Guideline input (mocap) trajectory for *Step up* motion

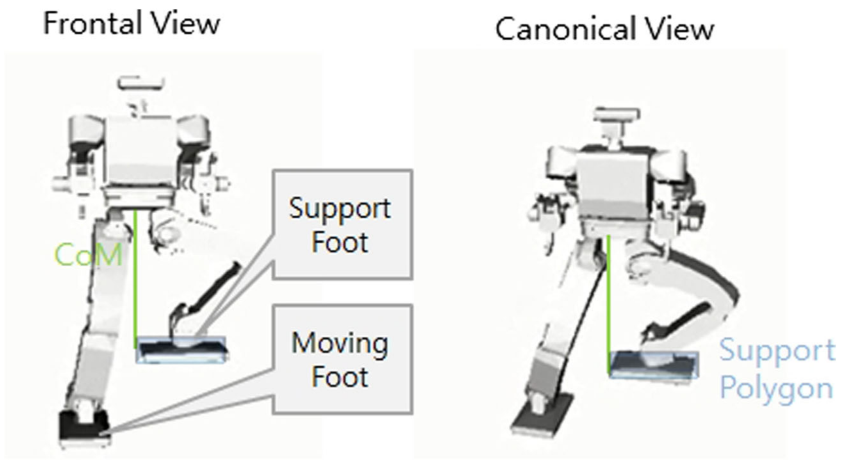


Fig. 14 Comparison between the support polygon and the CoM position

input trajectory. Through an integration process, the discrete frames were continuously merged to generate the DRC-Hubo's smooth motion.

In this study, the processing time for optimization (including post-processing) took between 212 and 255 seconds (Intel 1.8 GHz i-5 processors and 6 GB of RAM). From the initial guideline trajectory (31 to 32 seconds length), 800 neighbor samples were optimized per time step. The net result is that it took about 4 minutes to convert the raw captured motion of human subjects to the vehicle mounting trajectory which is optimized for DRC-Hubo. In similar studies, [17] generated the optimized trajectories for vehicle ingress motion in 7 minutes (Xeon 3.4 GHz processor and 8 GB of memory) and [18] took 20 minutes for optimizing the humanoid's whole body movements (Intel 1.70 GHz i5-2557M processor). With consideration that initial path-planning process took about 3 minutes, each approach took 4 and 17 minute for the optimization process only. It demonstrates that, under the system with similar specifications, the presented approach provide compatible performance in computation speed with other existing approaches.

4 Experimentation and Analysis with Vehicle Type 1: Golf Cart

In ingress, there exists two phases which can be considered as discrete states. First phase is *Step*: step up on vehicle floor, sit down and move to the driver's seat. Second phase is *Interface*: place a foot on an acceleration pedal and grab a wheel. For egress, phase *Step* can be replaced with *Step down*: move back, lift a body from seats and get both foot from floor down to ground. Toward task completion, each phase should be progressed.

In this paper, *Step* phase is mainly explored and presented. The *Step* phase consists of two different parts: *Step up* and *Sit down and Scoot*.

The initial guideline trajectories were produced from the captured motions of a human subject in mocap. The person was directed to step on the floor, to sit down on companion's seats first and then scoot to the driver's seat. The trajectory was then revised with respect to all the defined constraints during optimization process.

The processed trajectory (optimized motion) was initially tested using OpenHubo which is a virtual model of

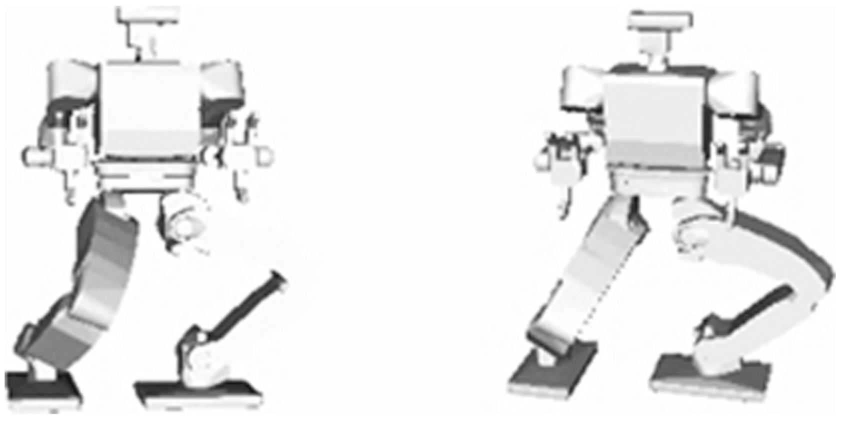


Fig. 15 Joint angle limit obedience caused by non-natural posture

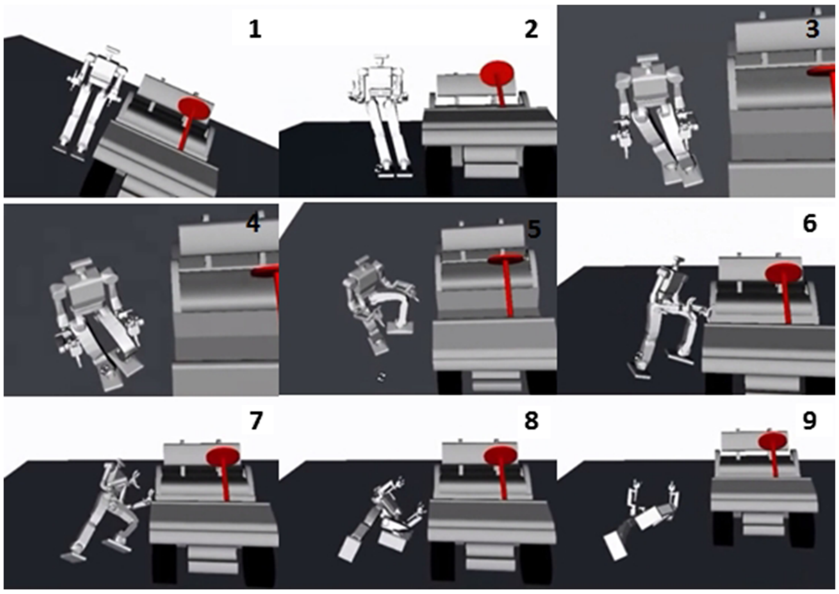


Fig. 16 Simulation result with raw trajectory

DRC-Hubo in an OpenRave simulation environment [21]. The toolkit provides motion control and planning of the DRC-Hubo model featuring physics based simulation, kinematic display and sensors. To check its physics and collisions, OpenHubo uses Open Dynamics Engine (ODE). It includes mass and inertia properties for DRC-Hubo as well as collision meshes.

Club Car (2008) is used for the initial test (Fig. 12). It is an electric powered utility cart and has a size of 87.6 cm × 97.8 cm. The KinFu module in the Point Clouds Library (PCL) is used to make the vehicle model in OpenHubo [22]. The module produced a high resolution surface model by combining multiple views from the RGBD camera.

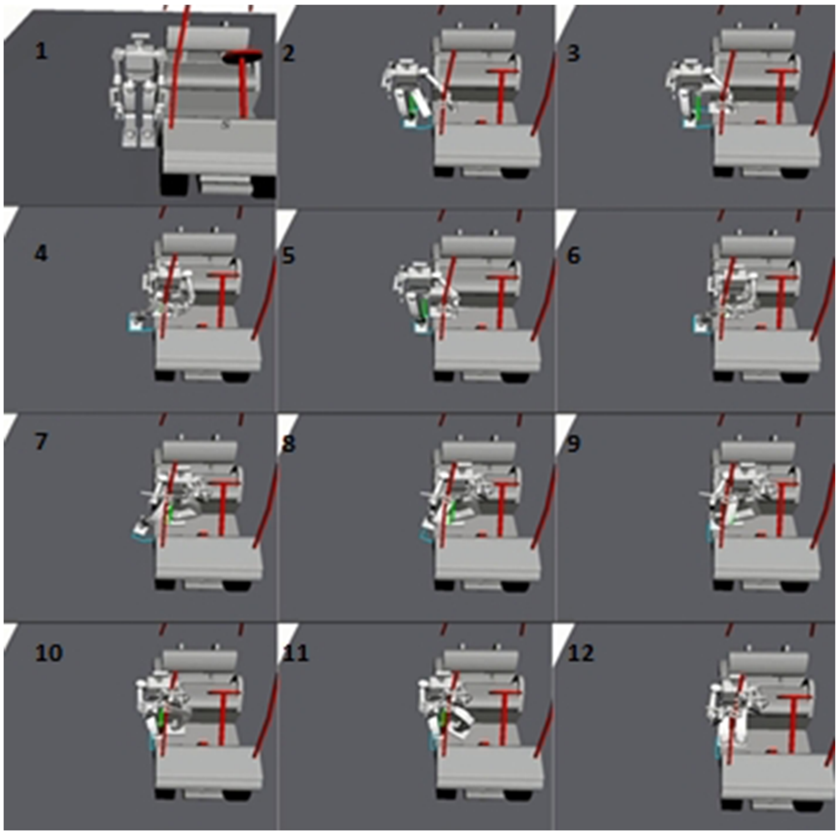


Fig. 17 *Step up* motion with the optimized trajectory

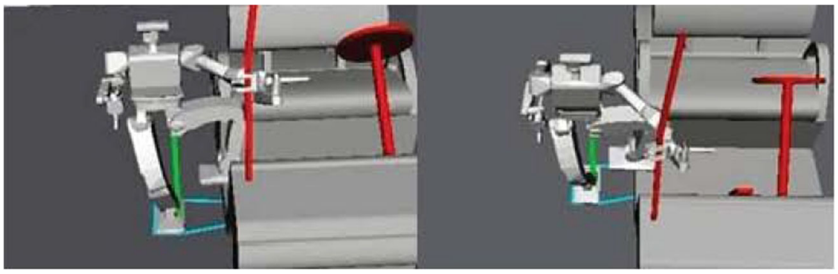


Fig. 18 Comparison between the support polygon and the CoM position

4.1 Guideline mocap Trajectory

Figure 13 shows the DRC-Hubo's motion (for the step-up part of *Step* phase) when the guideline (raw) input trajectory is applied. It is produced from the recorded human motions in mocap and is before the optimization process

Human drivers do not need to always place their CoM on the support polygon area to keep balancing. Therefore, as demonstrated in Fig. 14, the position of CoM often locates outside the polygon in the recorded movement.

Compared to Hubo+, the angle and torque limit values for each joint of DRC-Hubo are increased in its lower body. However, with the guideline mocap trajectory, DRC-Hubo still violates the joint limits in some cases (see the Fig. 15). Such unexpected postures can result in unnatural movement as well as high torque.

Figure 16 demonstrates the result of physics based simulation when the mocap trajectory (before optimization) is directly applied to DRC-Hubo model in OpenHubo. The trajectory did not guarantee collision avoidance and static balance of the robot. It resulted DRC-Hubo falls in beginning phase of *Step up*.

4.2 Test and Evaluation

Figure 17 shows the DRC-Hubo's for *Step up* motion when the optimized trajectory (generated from the trajectory optimization framework) is applied.

The result shows the opposite of the guideline input trajectory. With the optimized trajectory, the CoM position of DRC-Hubo stably lies inside the support polygon (see Fig. 18).

As observed in Hubo+'s case [13], when the optimized trajectory is applied, the DRC-Hubo also rotated its foot and changed the hip position to satisfy the given joint constraints. Figure 19 demonstrates that the DRC-Hubo rotated the right foot to keep the hip joint angle below its defined limits while not losing static balancing.

The trajectory optimization framework also generated other vehicle mounting movements for DRC-Hubo. Figure 20 demonstrates DRC-Hubo sitting in the passenger-seat (*Sit down*) and moving its body to the optimal driving position (*Scoot*). The net results is that DRC-Hubo completed the every parts of *Step* phase in ingress.

Similar to [13], only foot positions are designed and planned for end-effector trajectories to be optimized in this study. However, hand-contacts are also partially implemented to assist balancing of DRC-Hubo against unknown dynamic factors of the vehicle. During the vehicle mounting, DRC-Hubo grabbed and held the firm parts of the vehicle (which were manually chosen). Since the hip position is already determined in the optimized trajectory, IK process is simply implemented (between the upper-body of DRC-Hubo and the grabbed part) to generate the arm motion. By selecting different parts to grasp in the vehicle, multiple upper-body motions were also produced for DRC-Hubo. Figure 21 shows the DRC-Hubo's ingress with different hand-contacts.

Figure 22 shows the *Step down* motion of DRC-Hubo (during egress) after arrival.

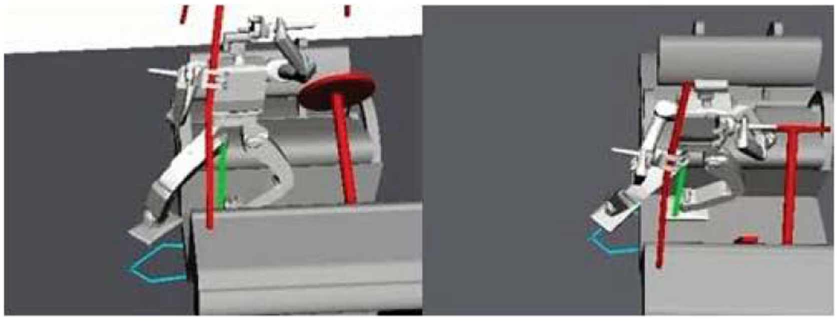


Fig. 19 Heading direction(yaw axis) change of the left foot in the optimized motion

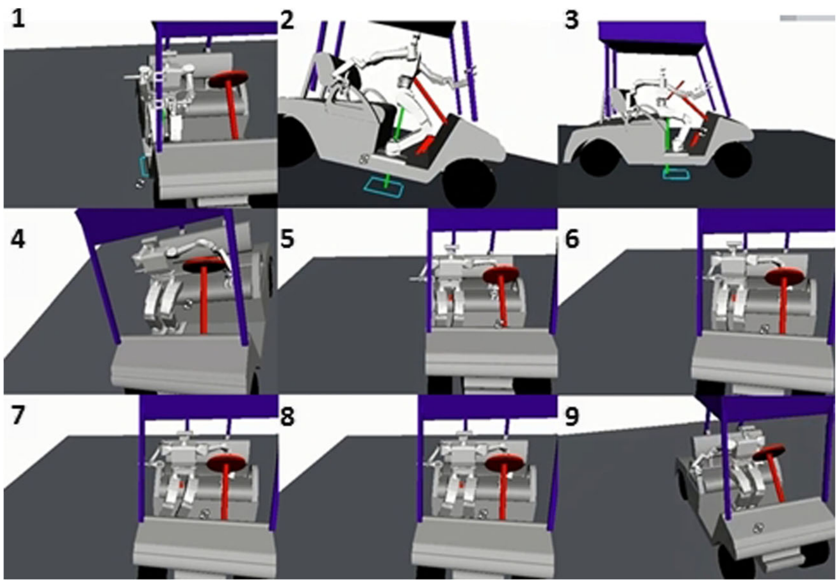


Fig. 20 *Sit down and Scoot* motion with the optimized trajectory

4.3 Verification

To confirm the efficacy of the presented approach, the experimental verification is also conducted with the real physical platform of the full-sized humanoid, DRC-Hubo. Figures 23 and 24 show the DRC-Hubo's *Step up* motions during ingress with the optimized trajectory.

Self-collision and joint limit issues did not happen in the experimentation using DRC-Hubo. In a previous study [13], Hubo+ made a few issues such as overheating and collisions during ingress.

Figure 25 demonstrates the DRC-Hubo's sitting and scooting to the driving position.

Figures 26 and 27 demonstrate the DRC-Hubo's *Step down* motions during egress.

5 Experimentation and Analysis with Vehicle Type 2: Polaris

Polaris Ranger XP 900 (2013) is used for the second test. The utility vehicle has a 4-stroke twin cylinder engine (gas

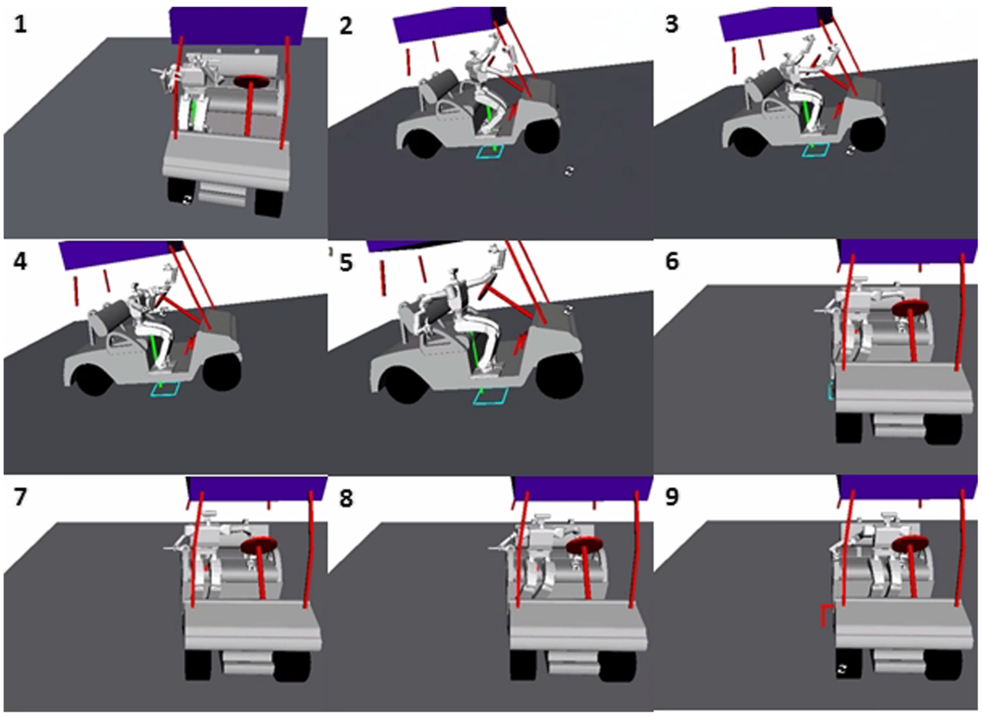


Fig. 21 *Sit down and Scoot* motion with different hand-contacts

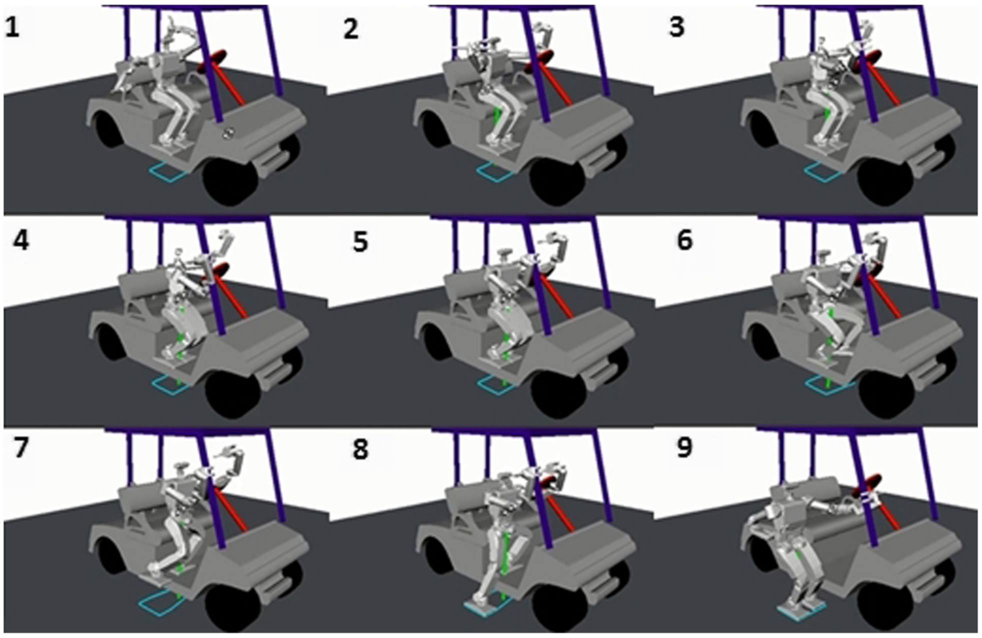


Fig. 22 *Step down* motion during egress with the optimized trajectory

powered) and a power steering wheel. It has a size of 296 cm × 152 cm × 193 cm. To make the virtual model in OpenHubo, the KinFu module is used again.

Figures 28 and 29 demonstrate the Polaris and its model in OpenHubo.

5.1 Test and Evaluation

For the experimentation, the guideline mocap trajectory which was used in Section 4.1 is reused. After optimization (considering the kinematic features of Polaris) process, the trajectory was tested in OpenHubo. Figures 30 and 31 show the DRC-Hubo's *Step up* motions during ingress with the Polaris model. Unlike the Club Car model in Section 4, the Polaris model did not have dynamic information (related to mass and inertia) of the vehicle. Therefore, the validity of the tested motion (such as collision between DRC-Hubo and the vehicle) was manually checked during the simulation test.

5.2 Verification

After the evaluation process above, the trajectory is verified with the real physical platform. Figures 32 and 33 show the DRC-Hubo's *Step up* motion on Polaris.

6 Discussion

6.1 Analytic Verification

To prove the validity of the proposed framework, an experimental scenario is built following this sequence: 1) DRC-Hubo first lowers its pelvis; 2) then sways hip to shift its pelvis position to the right side. Figure 34 shows the conducted motion.

Mocap was used to design the guideline trajectories for the motions. The movement was captured and recorded with 10 Hz sampling frequency. The hip trajectory (in each lateral (Y) and the vertical (Z) axis) and the joint angle trajectory (of the captured motion) is demonstrated in Figs. 35 and 36. As shown in Fig. 35, the hip movement is characterized as: -250 mm in Y axis and -150 mm in Z axis.

Mocap is a direct approach to design and plan motions in high dimensional spaces. From the past, recorded human motion data has been used for generating trajectories for various motion of robots, even real humanoids [10]. However, the trajectory recorded by mocap doesn't assure the robot's static balancing. Moreover, it often violates joint's kinematic constraints. The net result is that there exists a significant kinematic difference between the human body and humanoid. To overcome such limitations, the captured trajectory needs to be updated based on the required dynamic and kinematic constraints.

When various constraints for DRC-Hubo's trajectory are given like Eq. 2,

where

$$\left.\begin{aligned} q_{t0} = q_0, \dot{q}_{t0} = 0, \ddot{q}_{t0} = 0 \\ q_{tf} = q_0, \dot{q}_{tf} = 0, \ddot{q}_{tf} = 0 \\ q^- <= q_t <= q^+ \\ \dot{q}^- <= \dot{q}_t <= \dot{q}^+ \\ y^- <= y(q_t)_{pelvis} <= y^+ \end{aligned}\right\} \quad \text{Constraints} \tag{2}$$

the performance error of the captured motion is measured by the R^2 Norm [23]. q_t is the joint angles of the DRC-Hubo.

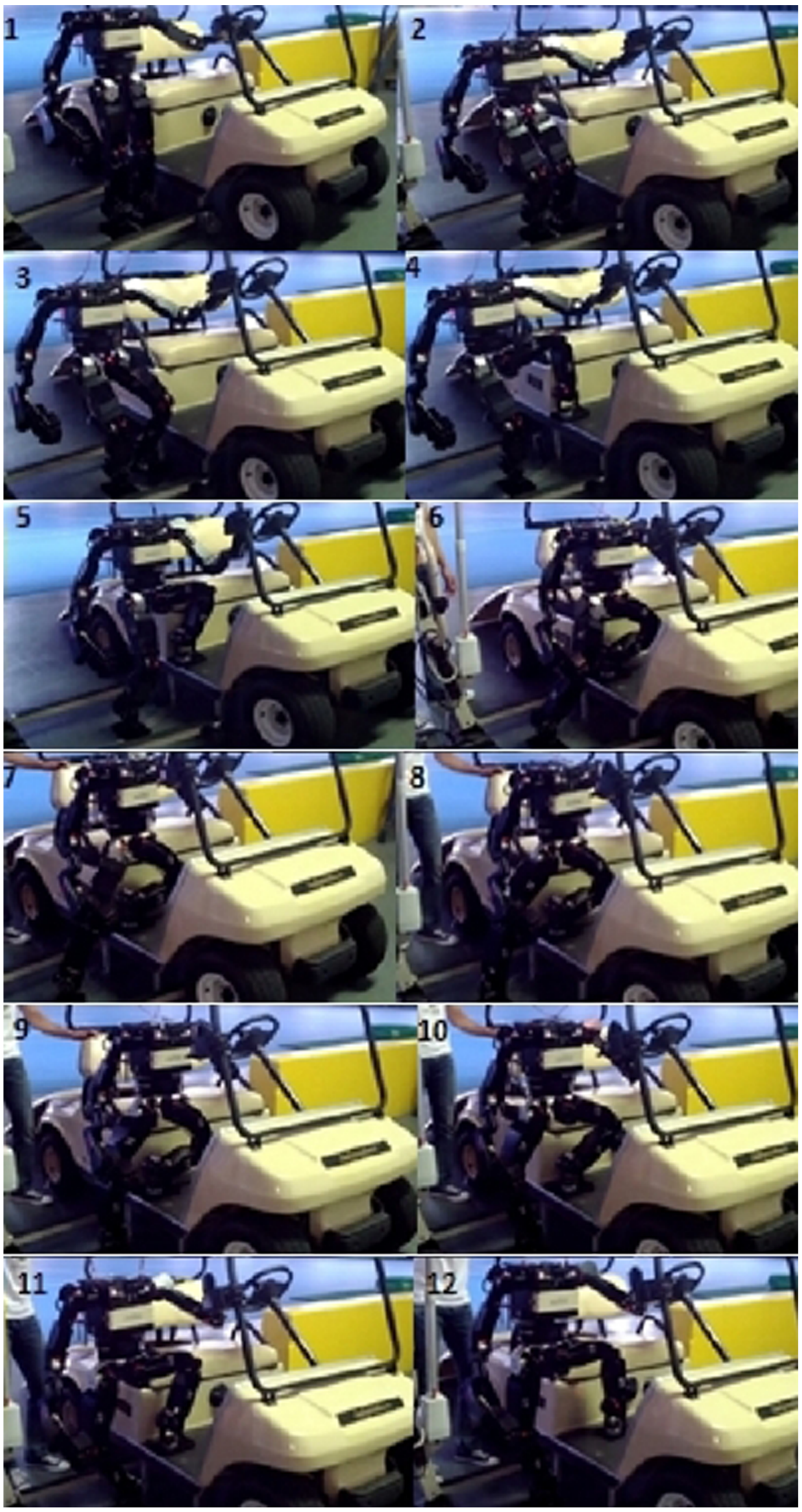


Fig. 23 DRC-Hubo's *Step up* motion (1st half phase)

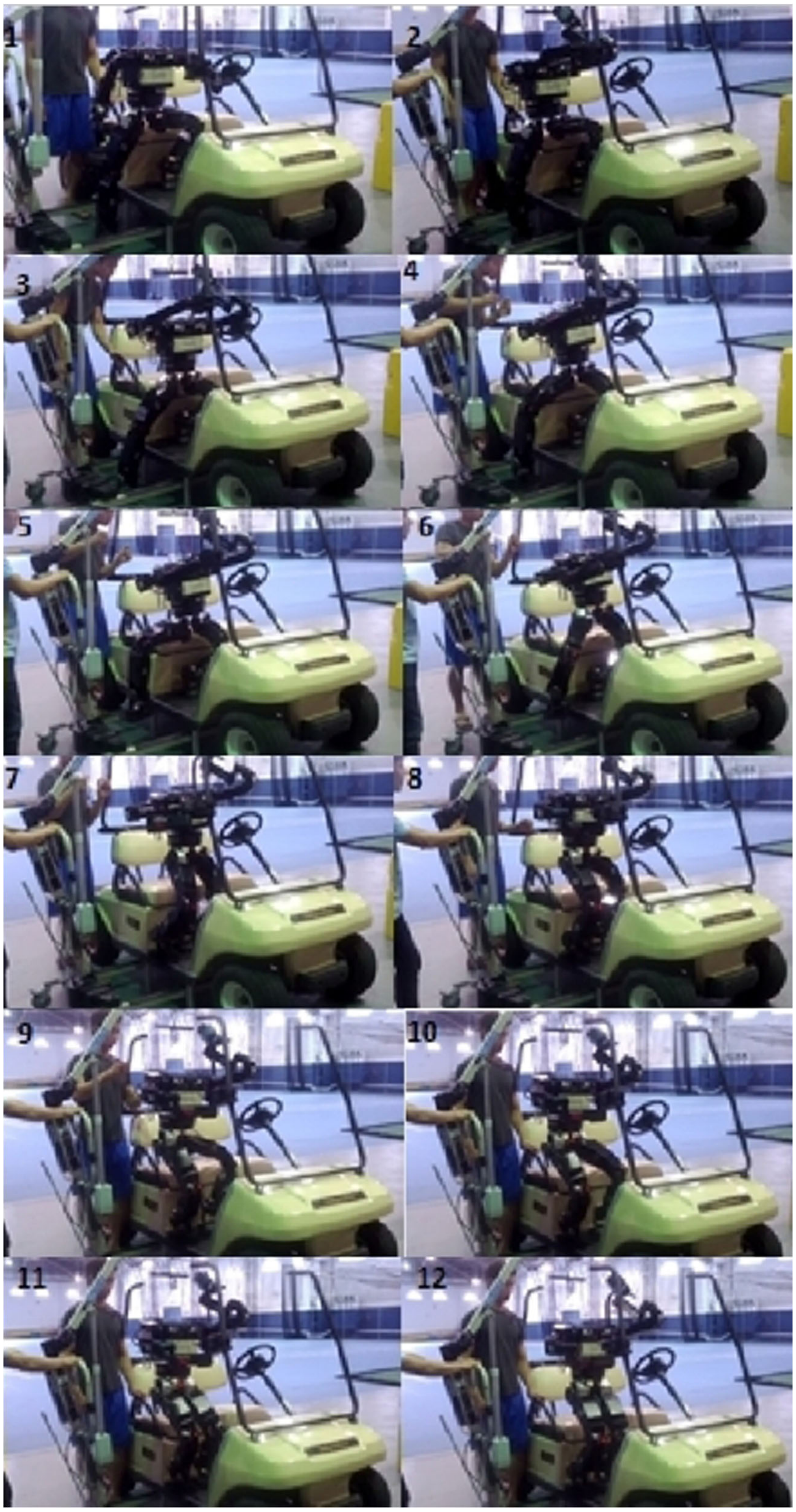


Fig. 24 DRC-Hubo's *Step up* motion (2nd half phase)

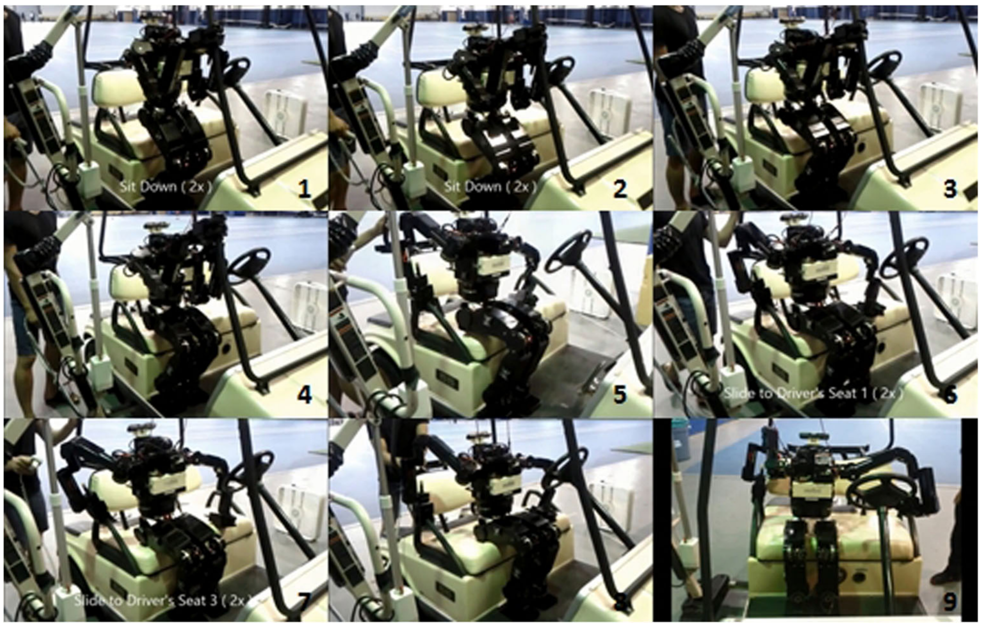


Fig. 25 DRC-Hubo's *Sit down and Scoot*

q^-, q^+, $\dot{q}^-$ and $\dot{q}^+$ are determined based on the robot's dynamic and kinematic limits of each joint (refer Figs. 10 and 11) and sampling time (100 ms). Bounds for $y(q_t)_{pelvis}$ is determined considering the supporting foot'width and its position.

The distance discrepancy between the pelvis position (from the recorded-motion) and its limits is computed as 2.722 mm. The error between each joint angle and its limit is also averaged as 4.998 degree. When 0.50 mm and 0.10 degree are chosen as minimum threshold costs, the

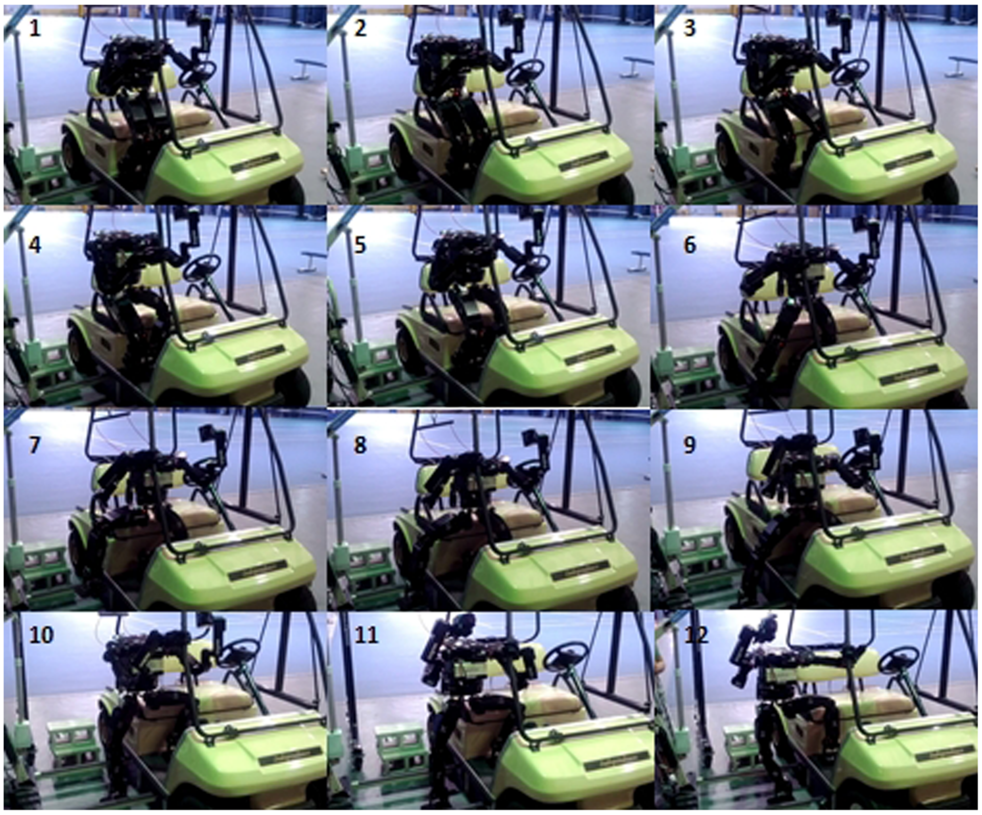


Fig. 26 DRC-Hubo's *Step down* motion during egress (1st half phase)

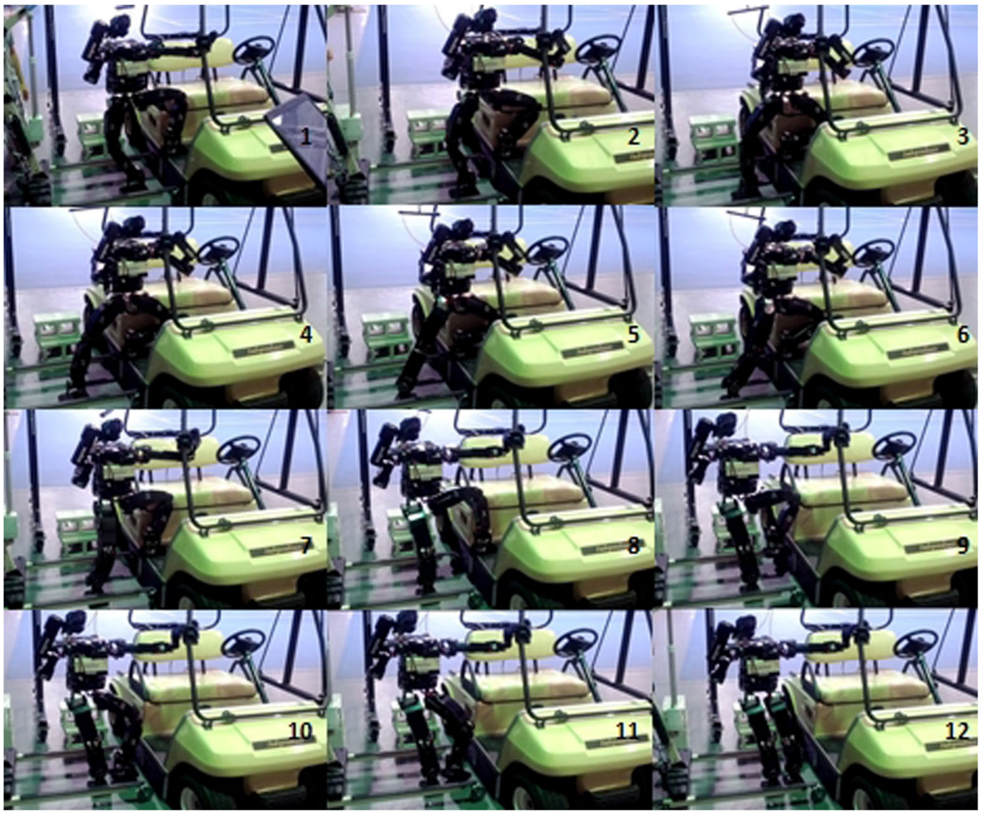


Fig. 27 DRC-Hubo's *Step down* motion during egress (2nd half phase)

recorded-motion trajectory was not acceptable for the use in DRC-Hubo. Furthermore, the knee pitch joint does not satisfy its required limits (65 degree which is chosen based on the kinematic structure of DRC-Hubo).

In this section, one of the most commonly used optimization approach, a Lagrange multiplier method, is applied first to optimize the captured motion with respect to kinematic and dynamic constraints.

The motion re-targeting problem is initially formulated as follows:

$$\underset{q_t, \dot{q}_t, \ddot{q}_t}{\arg\min} \int_{-t_0}^{t_f} ((q_t - {q_t}^c)^T (q_t - {q_t}^c) \tag{3}$$

$$+\sigma (P_t - {P_t}^c)^T (P_t - {P_t}^c))\, \mathrm{d}t$$

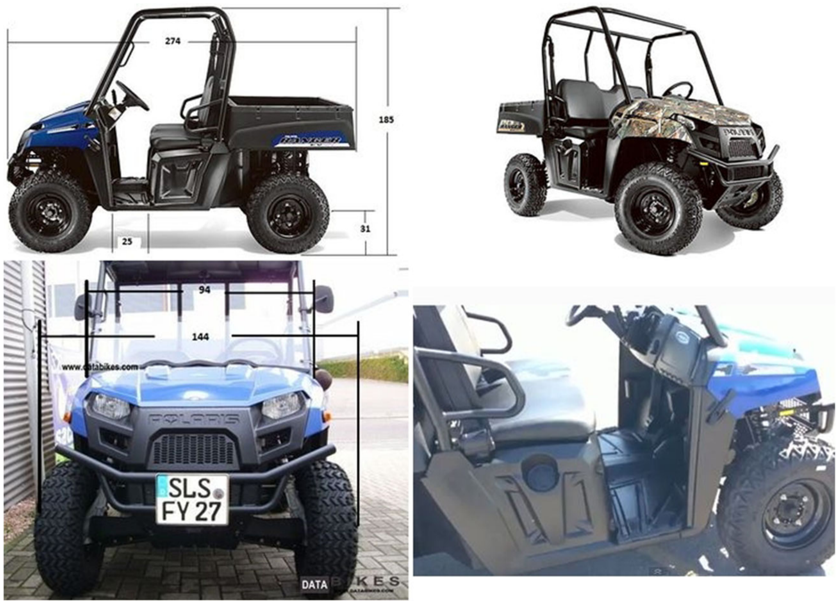


Fig. 28 Polaris Ranger XP 900

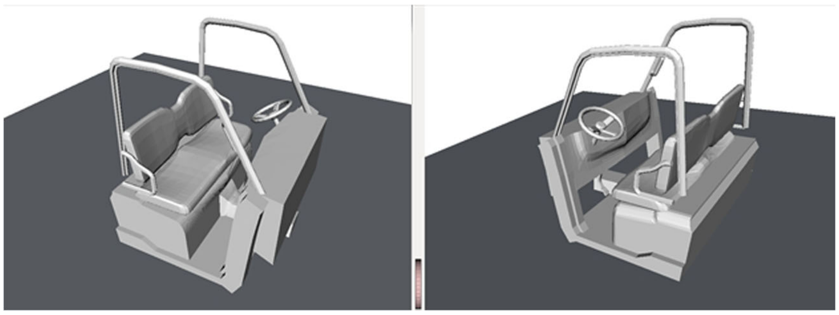


Fig. 29 Polaris model in OpenHubo

subject to Constraints

where σ is an user chosen constant. ${q_t}^c$ is the joint angles of the captured subject (in input trajectory). P_t and ${P_t}^c$ are the hip and end-effector positions (in Cartesian coordinate) of the DRC-Hubo and the subject.

The re-targeting problem can be transformed into a classical optimization problem as follows:

$$\arg\min_{X_t} L(X_t) \tag{4}$$

subject to

$$H(X_t) = 0$$

$$G(X_t) <= 0$$

with

$$X_t = [q_t, \dot{q}_t, \ddot{q}_t]^T$$

$$L(X_t) =$$

$$\int_{-t_0}^{t_f} \Big((q_t - {q_t}^c)^T (q_t - {q_t}^c) + \sigma (P_t - {P_t}^c)^T (P_t - {P_t}^c) \Big)\, \mathrm{d}t$$

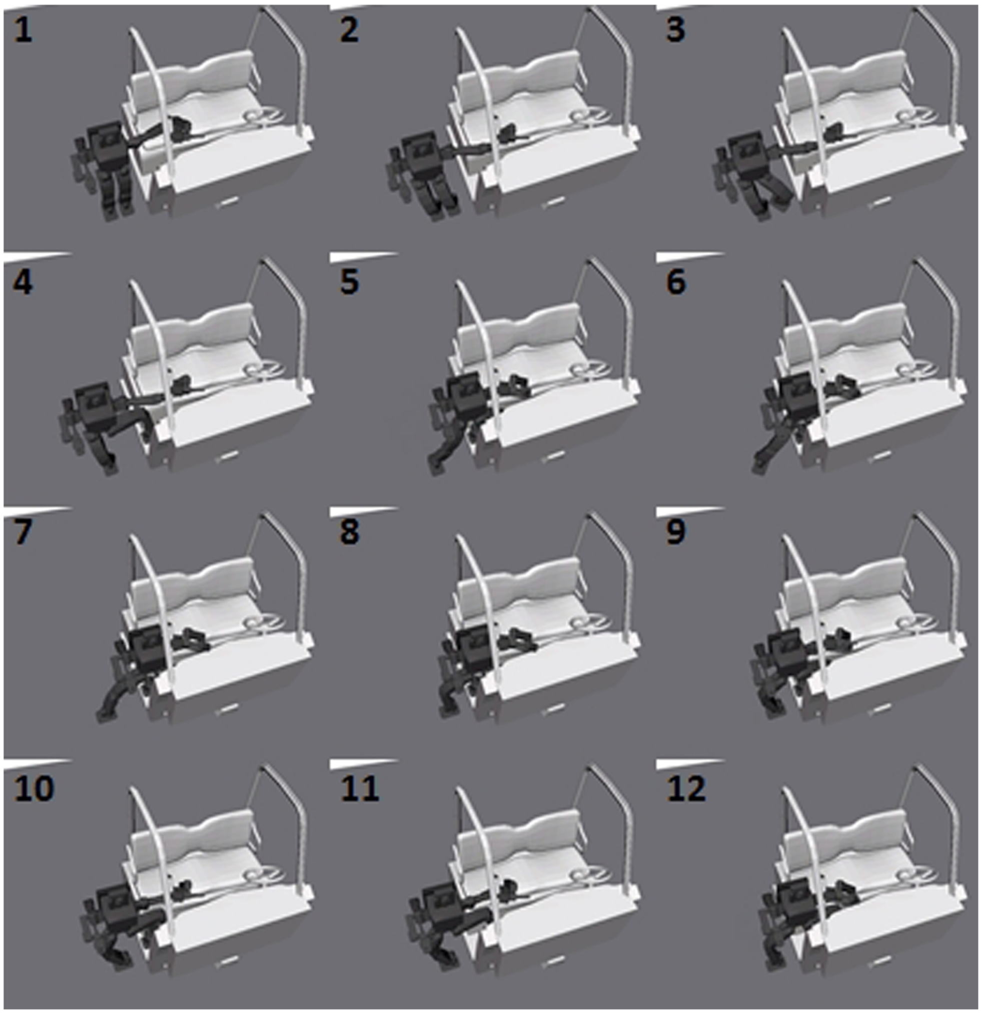
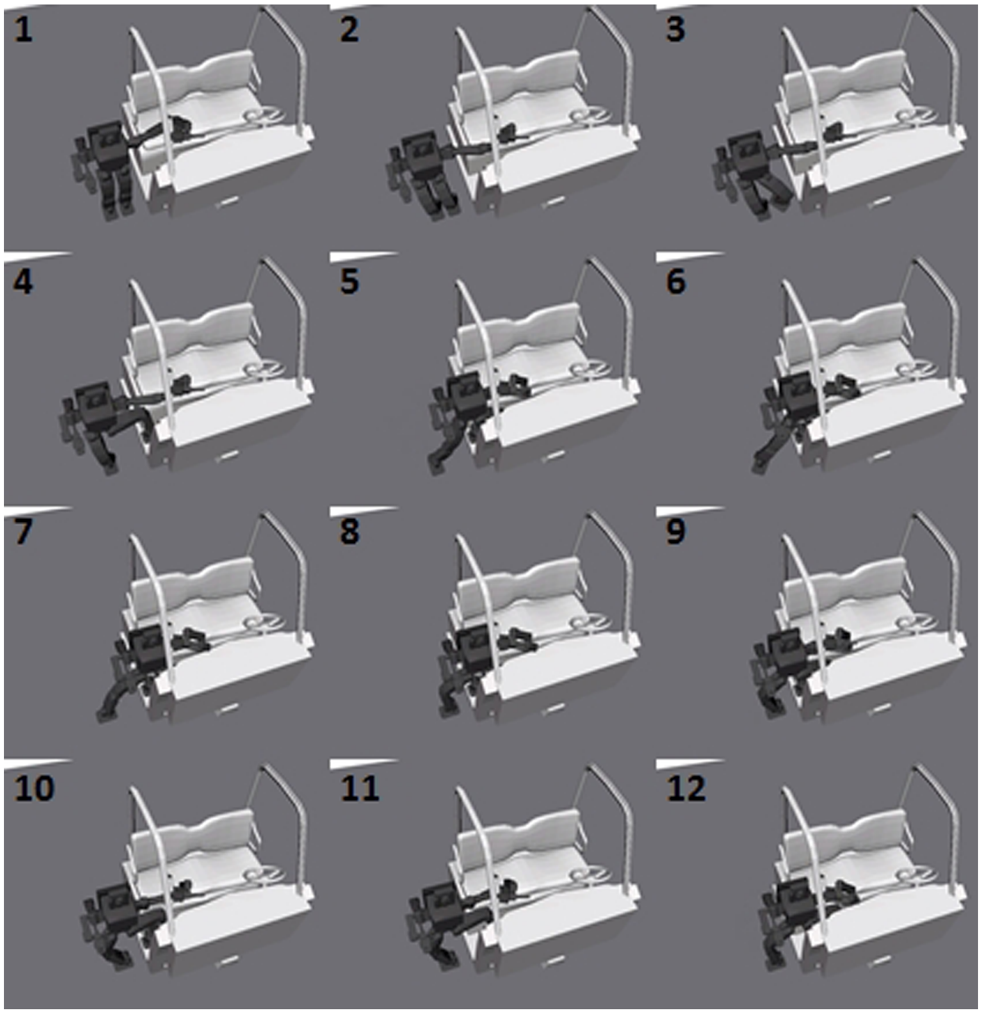


Fig. 30 The optimized trajectory for DRC-Hubo's *Step up* motion with Polaris (1st half phase)

Fig. 31 The optimized trajectory for DRC-Hubo's *Step up* motion with Polaris (2nd half phase)



$$G(X_t) = \begin{bmatrix} \dot{q}_t - \dot{q}^+ \\ -\dot{q}_t + \dot{q}^- \\ q_t - q^+ \\ -q_t + q^- \\ y(q_t)_{pelvis} - y^+ \\ -y(q_t)_{pelvis} + y^- \end{bmatrix}, H(X_t) = \begin{bmatrix} q_{t0} - q_0 \\ \dot{q}_{t0} \\ \ddot{q}_{t0} \\ q_{tf} - q_f \\ \dot{q}_{tf} \\ \ddot{q}_{tf} \end{bmatrix}$$

The problem is then transformed to the minimization of optimization problem with constraints (which can be solved with the Lagrange multiplier method). A Gauss-Newton method is used to solve the unconstrained optimization problem. Time derivatives of dynamic and kinematic costs are calculated by recursive way. Through tracing the configuration with the minimum value (of optimization problem), the final optimized trajectory is generated. A gradient descent update rule is applied for the minimization process.

The results from the Lagrange multiplier is demonstrated in Fig. 37. The distance discrepancy between the optimal pelvis and its limits is computed as 0.216 mm. The error between each joint angle and its limit is averaged as 0.004 degree. The error values of both pelvis and joint angle are decreased dramatically after the Lagrange multiplier process. They have both lower costs than the defined threshold. After the optimization process, the knee pitch joint's peak angle value is also decreased below its limit.

For analytical comparisons with techniques above (mocap and Lagrange multiplier), the same input trajectory (recorded motion) is assigned to the proposed learning based optimization framework. Weights on each cost function module were chosen considering the relative importance of the corresponding penalty. For this analysis, the joint angle limits and the pelvis position received the highest weights.

Figure 38 shows the generated results. The distance discrepancy between the optimal pelvis and its limits is computed as 0.220 mm. The error between each joint angle and its limit is averaged as 0.087 degree. Similar to the Lagrange multiplier, the error values of pelvis position and joint angle are reduced in the processed trajectories (lower costs than the defined threshold). The knee pitch joint's peak angle value is also reduced below its limit. The result proves that the presented framework

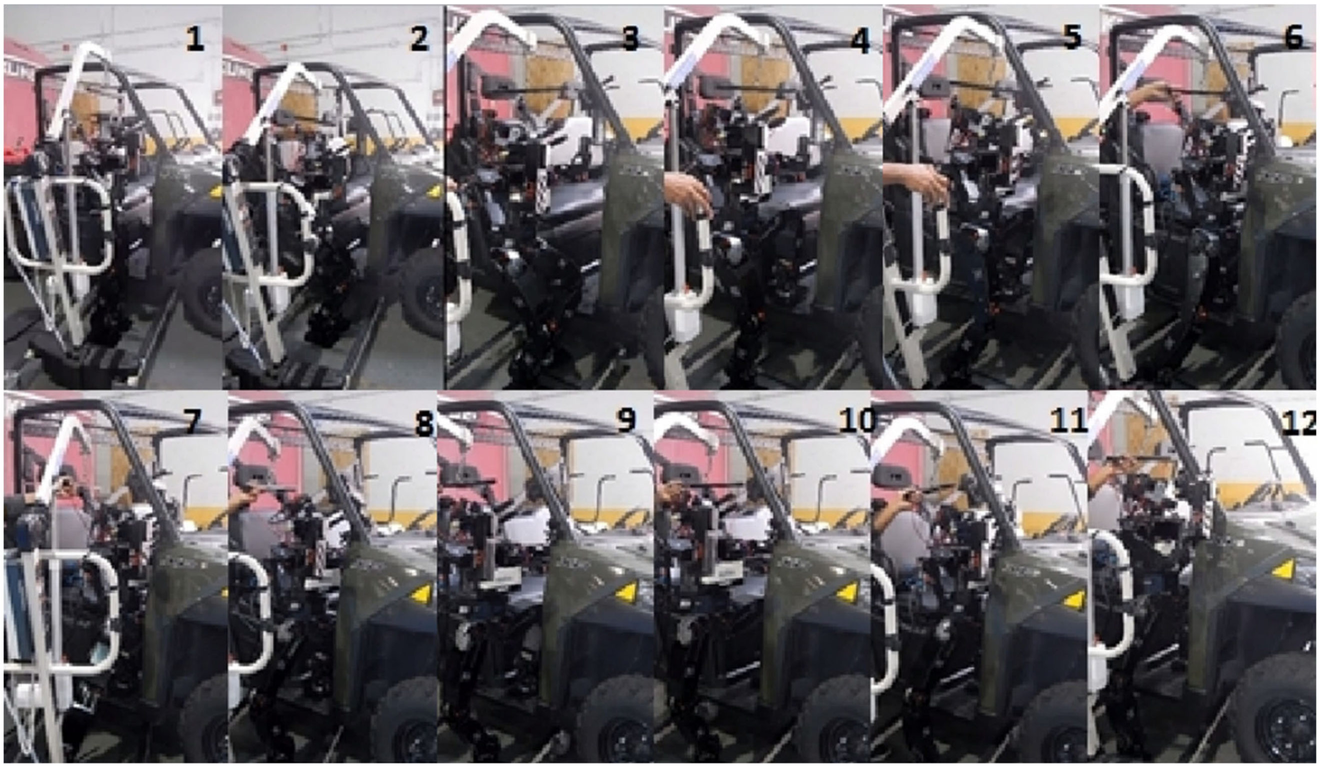


Fig. 32 DRC-Hubo's *Step up* motion with Polaris (1st half phase)

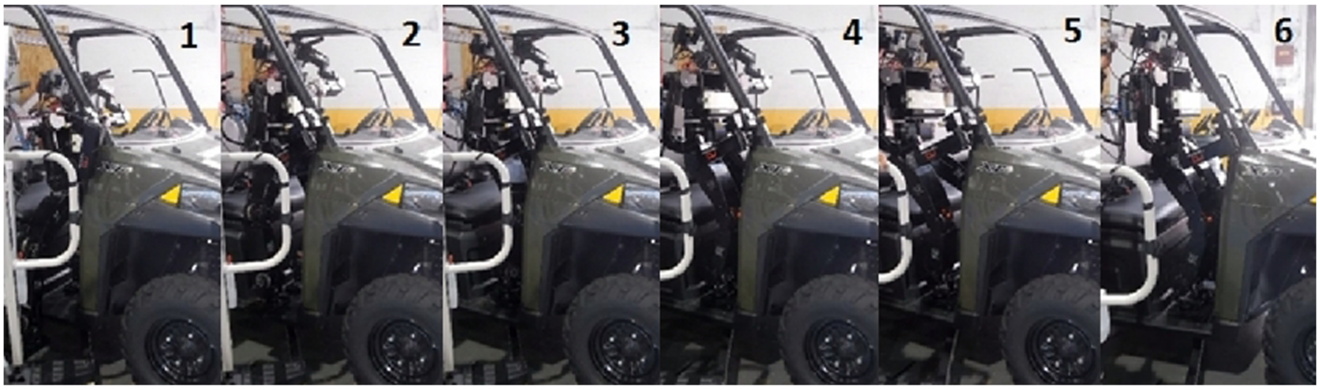


Fig. 33 DRC-Hubo's *Step up* motion with Polaris (2nd half phase)

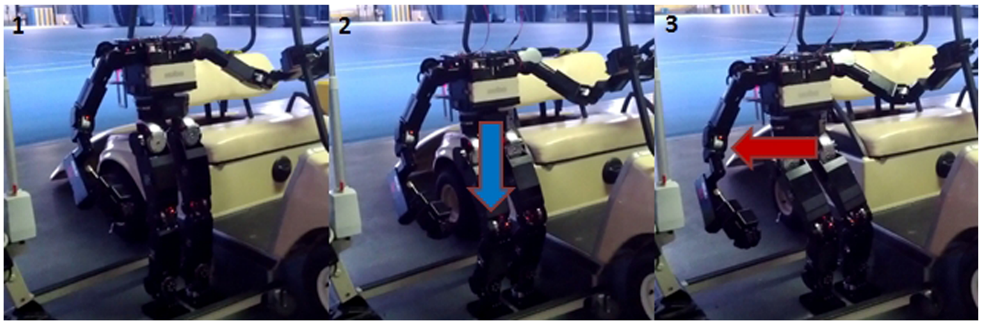


Fig. 34 Swaying movement of DRC-Hubo

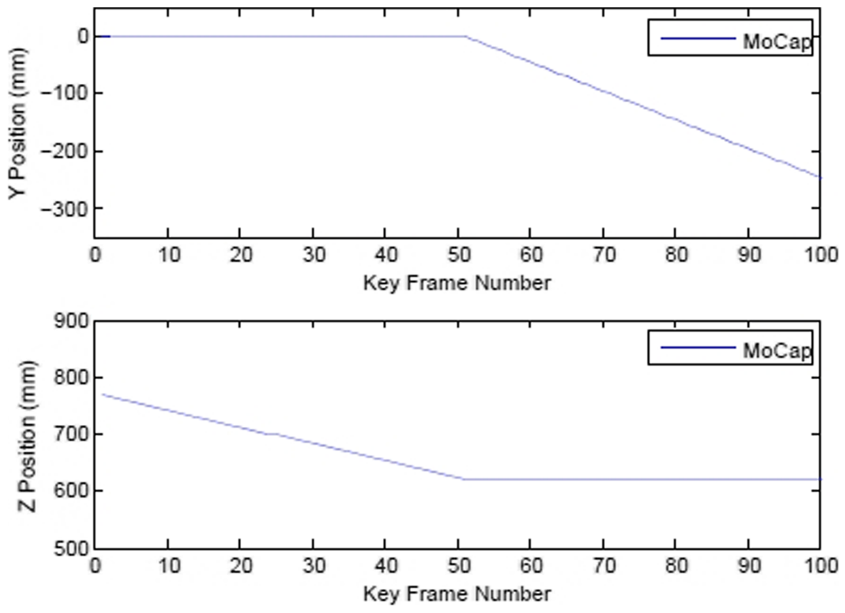


Fig. 35 Hip trajectory from the captured motion

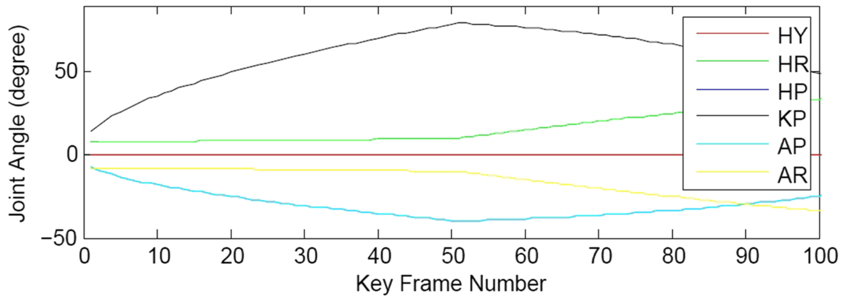


Fig. 36 Joint angle trajectory for right leg (HY: hip-yaw, HP: hip-pitch, HR: hip-roll, KP: knee-pitch, AP: ankle-pitch, AR: ankle-roll)

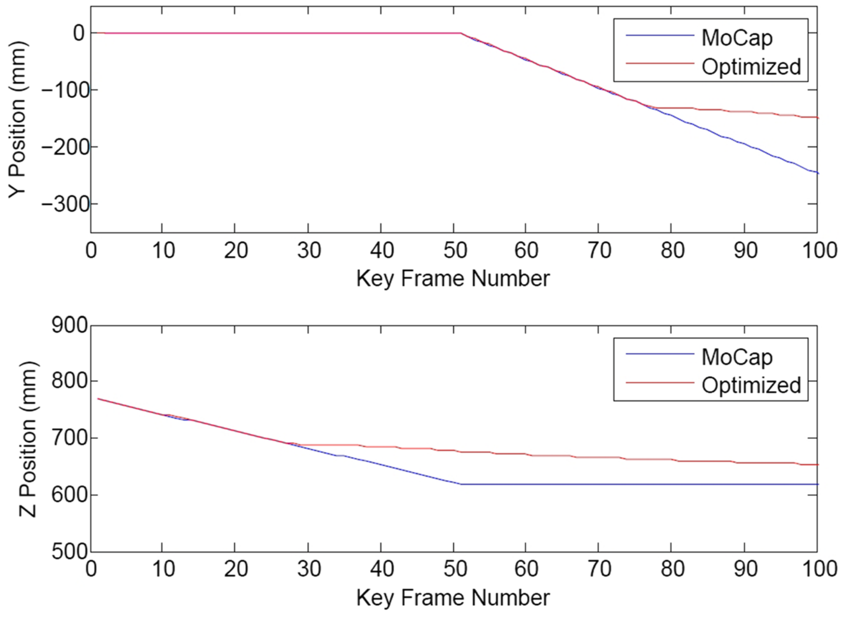


Fig. 37 Trajectories after the Lagrange Multiplier method

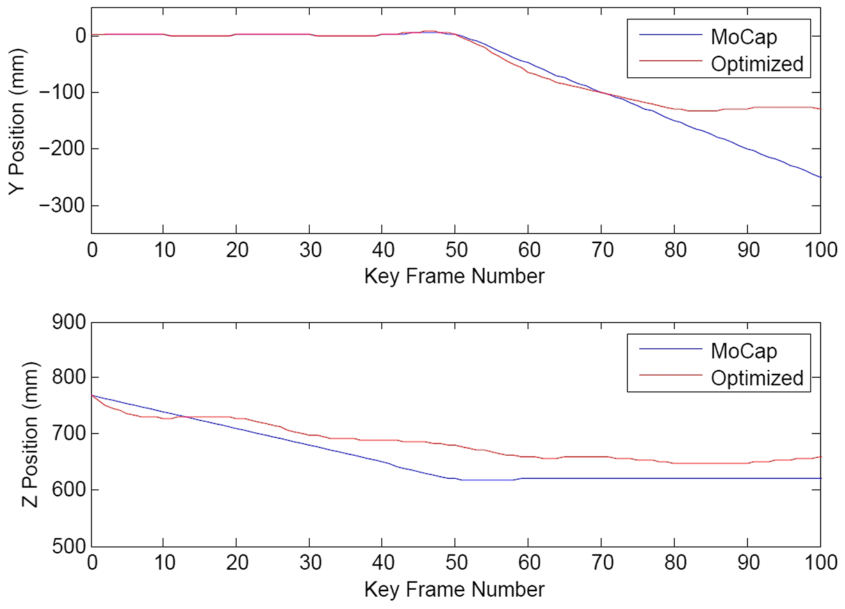


Fig. 38 Trajectories after reinforcement learning optimization

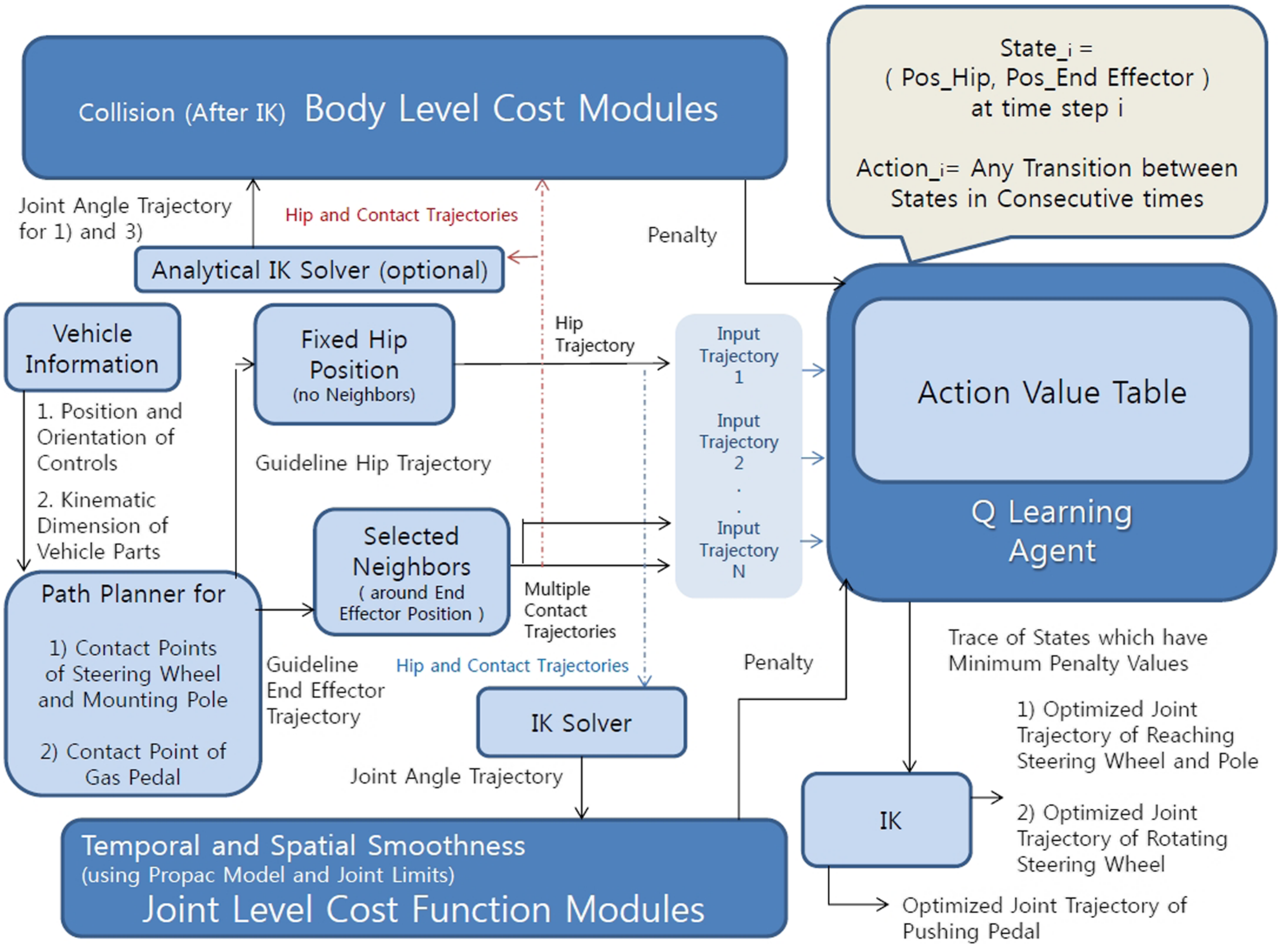

Collision (After IK) Body Level Cost Modules
State_i = (Pos_Hip, Pos_End Effector) at time step i
Action_i= Any Transition between States in Consecutive times
Joint Angle Trajectory for 1) and 3)
Hip and Contact Trajectories
Penalty
Analytical IK Solver (optional)
Vehicle Information
1. Position and Orientation of Controls
2. Kinematic Dimension of Vehicle Parts
Fixed Hip Position (no Neighbors)
Hip Trajectory
Input Trajectory 1
Input Trajectory 2
Input Trajectory N
Action Value Table
Q Learning Agent
Guideline Hip Trajectory
Selected Neighbors (around End Effector Position)
Multiple Contact Trajectories
Path Planner for
1) Contact Points of Steering Wheel and Mounting Pole
2) Contact Point of Gas Pedal
Guideline End Effector Trajectory
Hip and Contact Trajectories
Penalty
Trace of States which have Minimum Penalty Values
1) Optimized Joint Trajectory of Reaching Steering Wheel and Pole
IK Solver
Joint Angle Trajectory
IK
2) Optimized Joint Trajectory of Rotating Steering Wheel
Temporal and Spatial Smoothness (using Propac Model and Joint Limits)
Joint Level Cost Function Modules
Optimized Joint Trajectory of Pushing Pedal

Fig. 39 *Interface* phase trajectory optimization framework

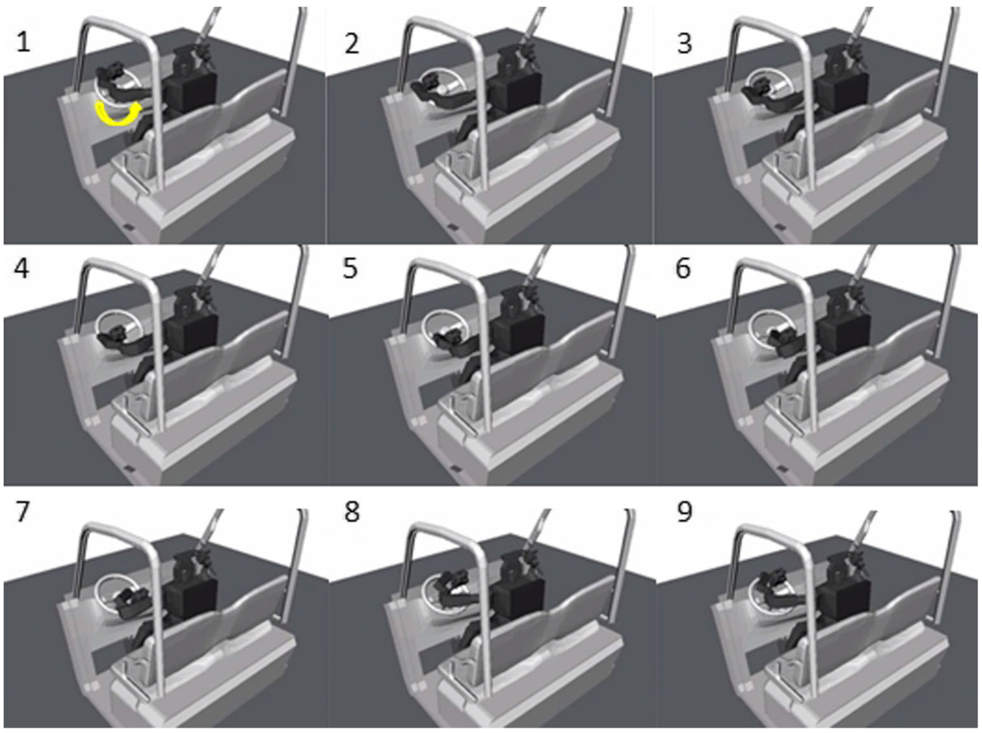


Fig. 40 Wheel-rotating motion (OpenHubo)

can optimize the trajectory under the defined set of constraints.

The reinforcement learning agent based trajectory optimization also conveys advantages over other techniques. In case of the Lagrange multiplier and Linear quadratic method [18], they provide the single optimal trajectory under the determined kinematic and dynamic limits.

However, the learning agent searches over all the possible cases (bins) in the Q table and records the cost for each case. Therefore, it can allow multiple solution trajectories to be produced (if they have costs below the defined threshold). The completed table can be re-used as needed under identical constraints.

From Sections 4.3, 5.2 and 6.1, the authors conclude that:

– The trajectory optimization framework optimized the input trajectory as regards all the given kinematic and dynamic constraints.
– The optimized trajectory can enable the full-sized humanoid to ingress the given vehicles.
– The Q value table contains multiple solution trajectories with different kinematic and dynamic features.

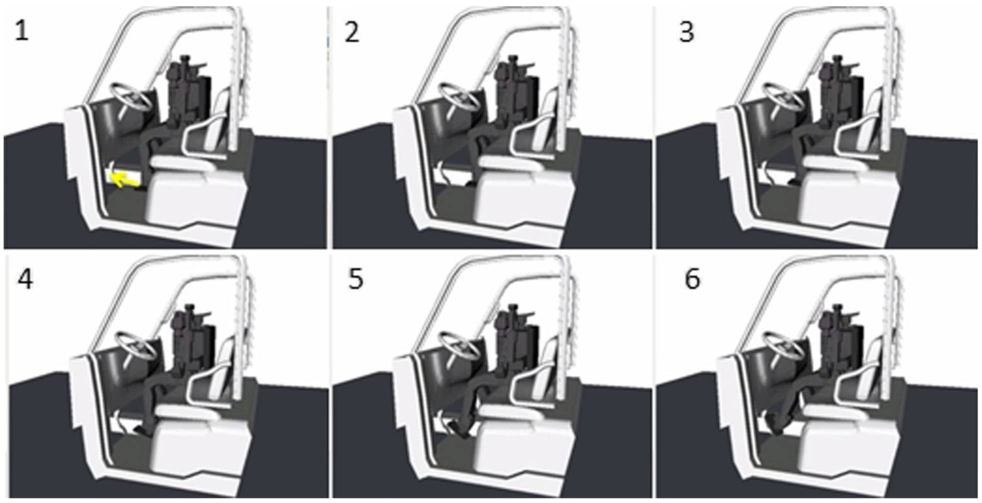


Fig. 41 Pedal-reaching motion (OpenHubo)

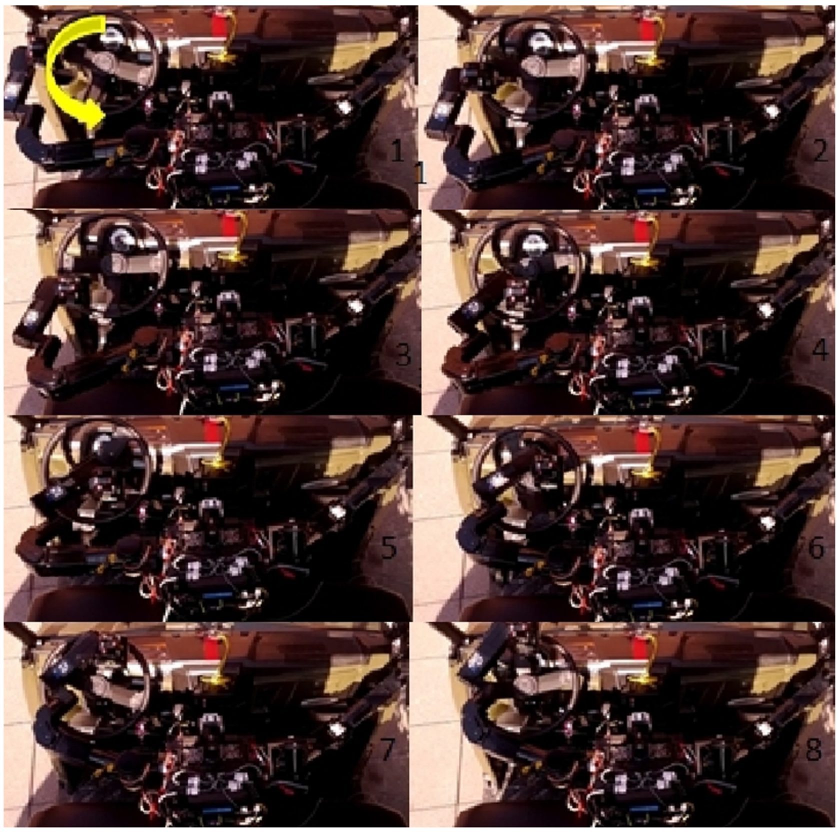


Fig. 42 DRC-Hubo's wheel-steering motion

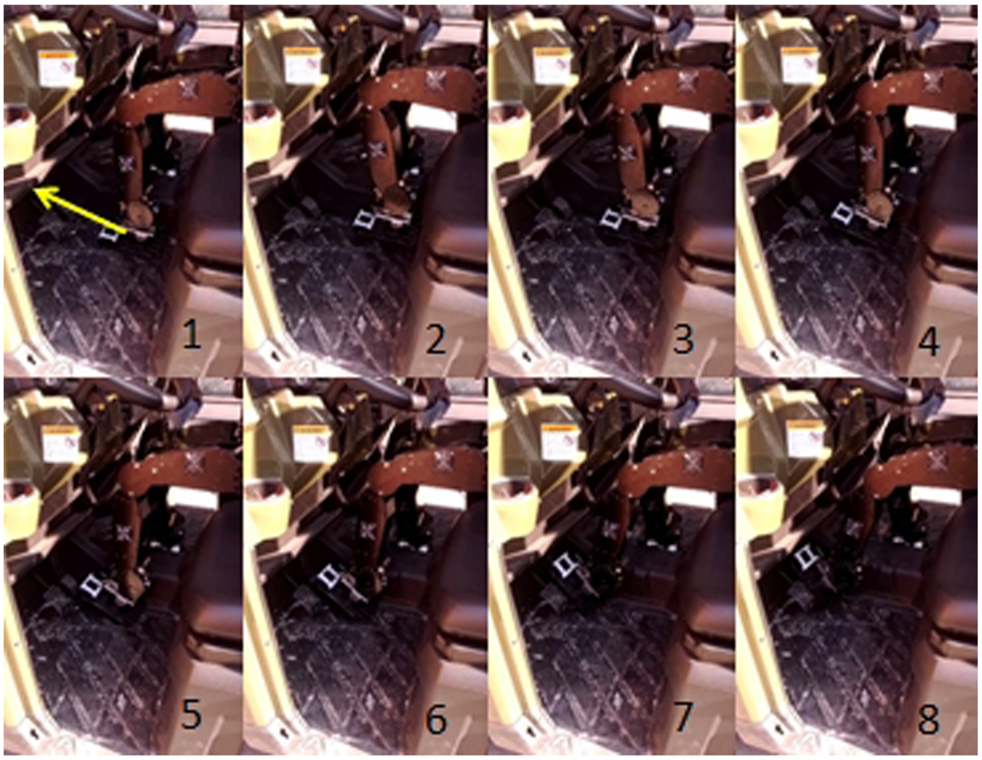


Fig. 43 DRC-Hubo's pedal-reaching motion

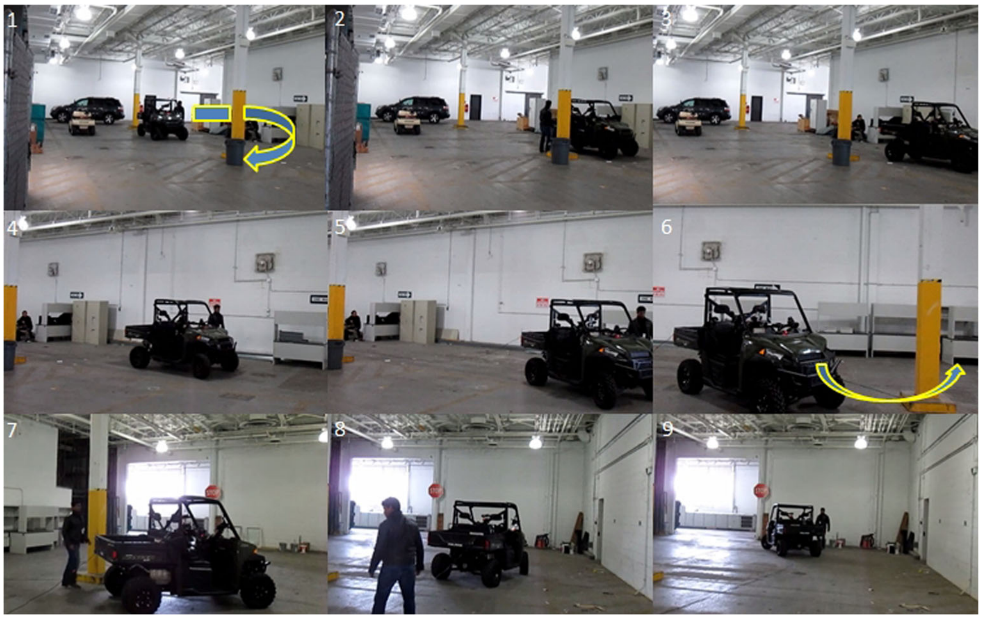


Fig. 44 DRC-Hubo's Polaris driving in a parking garage

6.2 Extended Study: *Interface* phase

This section briefly presents an extended study of the trajectory optimization framework for the second phase (*Interface*) of vehicle mounting task. During the phase, the robot should be able to actuate all control inputs of the given vehicle. For this, implementation of two different tasks were conducted: 1) reaching and rotating the steering wheel; 2) reaching and pushing the control pedal.

While the main structure of the optimization framework (which is used for *Step* phase)is kept, the shortest paths (between goals and end-effectors) are used for initial motions to be processed. The trajectories were then optimized as regards the robot's kinematic and dynamic limitations and collision avoidance. Figure 39 illustrates the trajectory optimization framework for *Interface*phase.

Similar to the studies in *Step* phase, the experimental evaluation was first implemented through the simulation (with DRC-Hubo model in OpenRAVE). Figure 40 demonstrates the left arm's wheel-rotating motion and Fig. 41 shows the left foot's reaching motion on the gas pedal.

The effectiveness of the presented approach is verified in experiments with the physical platforms. Figures 42 and 43 show the DRC-Hubo's wheel-steering and pedal-reaching motion in Polaris.

More details with a building procedure of trajectory optimization framework for *Interface* phase are given in [12]. Compared to [24] which focuses on the relationship between the robot's body frame and the vehicle's control input, it optimizes the motions with respect to various constraints such as collisions, kinematic limitations and spatial smoothness.

Figure 44 demonstrates the DRC-Hubo's driving in the mock-up site. The replicas of the driving courses (each for Trials and Final) were built respectively in the indoor and outdoor parking lot. In the site test, DRC-Hubo successfully finished its driving within 14 minutes and 1 minute (average from 12 runs) for each course. DRC-Hubo demonstrated its stable and robust driving skill in DRC-Finals [6, 25].

7 Conclusion

This paper starts with broad impacts and various merits of the humanoid's vehicle handling capability. To enable the robot to mount vehicles, a whole body motion optimization framework is built under the multiple time-varying constraints. The framework transformed the guideline input trajectory to the humanoid's motion which meets various dynamic and kinematic requirements at the robot's body and joint levels. Experiments with the full-sized humanoid, DRC-Hubo, and analytical comparisons with other techniques verified the proposed approach. The optimization framework also showed its adaptability for designing other types of motions such as vehicle interfacing. The presented approach was continuously used for the team DRC-Hubo's driving task in DARPA Robotics Challenge.

References

1. Robot Olympics: 17 Cyborg Athletes to Vie for Glory in DARPA Challenge, livescience (2013)
2. DARPA's Rescue-Robot Showdown: Robots face off in first phase of DARPA's Robotics Challenge, IEEE Spectrum (2013)
3. http://www.darpa.mil, Tactical Technology Office, DARPA (2012)
4. http://archive.darpa.mil/roboticschallengetrialsarchive/, Darpa Robotics Challenge Trials 2013, DARPA (2015)
5. http://www.theroboticschallenge.org/, Darpa Robotics Challenge Finals 2015, DARPA (2015)
6. Sofge, E.: Robots are terrible at driving cars, popular science (2015)
7. WPI Robotics Team Finishes in 7th Place in National Competition, Worcester polytechnic institute news (2013)
8. Hasunuma, H., Nakashima, K., Kobayashi, M., Mifune, F., Yanagiharat, Y., Ueno, T., Ohya, K., Yokoi, K.: A Tele-operated Humanoid Robot Drives a Backhoe. In: IEEE International Conference on Robotics and Automation (ICRA), vol. 3, pp. 2998–3004, Taipei, Taiwan (2003)
9. Kavraki, L.: Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. In: IEEE Transactions on Robotics and Automation, 12-4, pp. 566–580 (1996)
10. Matsui, D., Minato, T., MacDorman, K.F., Ishiguro, H.: Generating Natural Motion in an Android by Mapping Human Motion. In: IEEE International Conference on Intelligent Robots and Systems (IROS) pp. 3301–3308, Alberta, Canada (2005)
11. Sohn, K., Oh, P.: Applying human motion capture to design energy-efficient trajectories for miniature humanoids. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Algarve, Portugal, pp. 3425–3431 (2012)
12. Sohn, K.: Optimization of Humanoid's Motions under Multiple Constraints in Vehicle-Handling Task, PhD Thesis, Department of Mechanical Engineering and Mechanics. Drexel University, PA (2014)
13. Sohn, K., Oh, P.: Optimization of Humanoid's Motions under Multiple Constraints in Ingress Task. In: Journal of Intelligent Service Robotics (ISR), pp. 1–18. Springer (2015)
14. Diankov, R.: Automated Construction of Robotics Manipulation Programs, PhD Thesis, Robotics Institute, Carnegie Mellon University (2010)
15. Oh, J.: DRC-Hubo (http://hubolab.kaist.ac.kr/p_drchubo). Critical Design Review of DARPA Robotics Challenge Trials 2013, Philadelphia, USA (2013)
16. Berenson, D., Srinivasa, S.S., Ferguson, D., Collet, A., Kuffner, J.: Manipulation planning with workspace goal regions. In: IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, pp. 618–624 (2009)
17. Qiu, Z., Escande, A., Micaelli, A., Robert, T.: A Hierarchical Framework for Realizing Dynamically-stable Motions of Humanoid Robot in Obstacle-cluttered Environments. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), Osaka, Japan, pp. 867–874 (2012)
18. Bouyarmane, K., Vaillant, J., Keith, F., Kheddar, A.: Exploring Humanoid Robots Locomotion Capabilities in Virtual Disaster Response Scenarios. In: 12th IEEE-RAS International Conference on Humanoid Robot (Humanoids), Osaka, Japon, pp. 337–342 (2012)
19. Watkins, C.J.C.H.: Learning from Delayed Rewards. PhD thesis, Cambridge University (1989)
20. Kwatny, H.G., Blankenship, G. Nonlinear Control and Analytical Mechanics: A Computational Approach (Control Engineering), 1edition. Birkhauser, Boston (2000)
21. Ellenberg, R., Oh, P.: Contact Wrench Space Stability Estimation for Humanoid Robots, IEEE International Conference on Technologies for Practical Robot Applications (tePRA), Woburn, MA, pp. 1–6 (2014)
22. Rusu, R., Cousins, S.: 3D is here: point cloud library (PCL), IEEE International Conference on Robotics and Automation (ICRA), pp. 1–4 (2011)
23. Deza, E., Deza, M.M.: Encyclopedia of Distances. Springer, Berlin (2009)
24. Paolillo, A., Gergondet, P., Cherubini, A., Vendittelli, M., Kheddar, A.: Autonomous car driving by a humanoid robot. J. Field Rob. **35**(2), 169–186 (2017)
25. Whitaker, I.: UNLV comes in eighth in international robotics competition, Las Vegas Sun (2015)

Kiwon Sohn received B.S. and M.S. degree of electrical engineering from Kyungpook National University (Daegu, South Korea) and University of Pennsylvania (PA, USA) in 2005 and 2007. He received Ph.D. degree of mechanical engineering from Drexel University (PA, USA) in 2014. He served as a Chief of Engineering in team DRC-HUBO@UNLV (the finalist of DARPA Robotics Challenge Finals) and DASL (Drones and Autonomous Systems Lab) in University of Nevada, Las Vegas between 2014 and 2016. Currently, he is working in University of Hartford as an assistant professor.

Prof. Sohn is a member of IEEE Robotics and Automation society.